

EXHIBIT 21

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Balassanian, et al.

Reexamination Control No.: 95/000,659

U.S. Patent No.: 6,629,163

Reexamination Request Filed: Feb. 13, 2012

For: METHOD AND SYSTEM FOR
DEMULPLEXING A FIRST
SEQUENCE OF PACKET
COMPONENTS WHEREIN
SUBSEQUENT COMPONENTS ARE
PROCESSED WITHOUT RE-
IDENTIFYING COMPONENTS

Examiner: Salman Ahmed

Technology Center/Art Unit: 3992

RESPONSE

Attn: Mail Stop "Inter Partes Reexam"
Central Reexamination Unit
Office of Patent Legal Administration
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed April 3, 2012, please enter and consider the following Remarks showing that the three claims under reexamination are patentable.

The pending claims are reflected in the Listing of Claims attached as an Appendix. No amendments have been made to the claims. Exhibits 1-2 are attached to this Response. The Patent Owner does not believe that extensions of time or other fees are required. However, if any fees are necessary to prevent abandonment of this reexamination, then such fees are hereby petitioned and hereby authorized to be charged to our Deposit Account No. 504592.

establish a *prima facie* case of obviousness because the above statements are both technically incorrect and do not follow the strict legal requirements set forth in Section V.B, above.

As an initial matter, the obviousness law requires that a prior art reference must be considered in its entirety, including portions that would lead away from the claimed invention. *W.L. Gore & Assoc., Inc. v. Garlock, Inc.*, 721 F.2d 1540, 1552-53 (Fed. Cir. 1983). Decasper discloses an architecture for an IP routing system. An IP router and its components, by definition, are only designed to handle packets in the IP (Internet Protocol) format—a single format. Accordingly, the Decasper architecture assumes that all packets will be in IP format, and therefore that the input and output of every component will be a packet with an IP format. Given the single format of data (IP) that Decasper processes, it teaches away from considering format compatibility when “dynamically identifying a non-predefined sequence of components.”

Nor would there be any “reason”—as required by *KSR*—to consider format compatibility in Decasper. The architecture is specifically designed to handle one format (IP), and thus, one of ordinary skill would be unconcerned about matching different formats. In fact, adding functionality to check or compare the output format of a packet processed by one IP component with the input format required by the next IP component, would add unnecessary and costly overhead. In short, no skilled person looking at Decasper would add the claim requirement.

Not only is there no reason to add format compatibility to Decasper, the Office Action fails to cite to a single piece of evidence for adding such a feature. See Office Action at 21. There is no cited patent or printed publication or any other document; instead, there are only bald assertions without the evidentiary support required by the law.

Moreover, those assertions are technically inaccurate. As noted above, the Examiner states that “it was well-known to those of ordinary skill in the art that certain operations of a

generated by performing the processing of a component for a packet is available to the component when the component processes the next packet of the message.” Claim 35 requires that “for each packet of each message, invoking the identified non-predefined sequence of components in sequence to perform the processing of each component for the packet wherein each component saves message-specific state information so that that component can use the saved message-specific state information when that component performs its processing on the next packet of the message.” For the same reasons set forth above in Section VII.A.3 with respect to claim 1, Kerr also fails to disclose these state information requirements in claims 15 and 35.

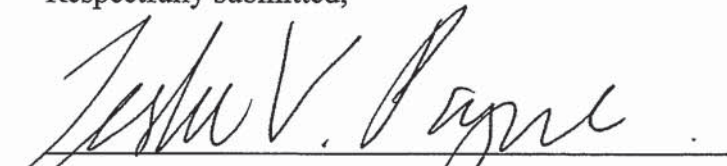
VIII. SECONDARY INDICIA EVIDENCE OF NON-OBVIOUSNESS

In addition to the above arguments showing that claims 1, 15 and 35 are non-obvious, Implicit presents the declaration of the inventor, attached as Exhibit 1. The declaration and the exhibits attached to the declaration further show that the claims are non-obvious.

IX. CONCLUSION

For the reasons set forth above, Patent Owner Implicit respectfully requests the PTO to withdraw its rejections of claims 1, 15, and 35 and to confirm the patentability of all claims.

Respectfully submitted,



Leslie V. Payne (Reg. No. 38,267)

EXHIBIT 22

1 SPENCER HOSIE (CA Bar No. 101777)
shosie@hosielaw.com
2 GEORGE F. BISHOP (CA Bar No. 89205)
gbishop@hosielaw.com
3 DIANE S. RICE (CA Bar No. 118303)
drice@hosielaw.com
4 HOSIE RICE LLP
Transamerica Pyramid, 34th Floor
5 600 Montgomery Street
San Francisco, CA 94111
6 (415) 247-6000 Tel.
7 (415) 247-6001 Fax

8 *Attorneys for Plaintiff*
9 *IMPLICIT NETWORKS, INC.*

10
11 UNITED STATES DISTRICT COURT
12 FOR THE NORTHERN DISTRICT OF CALIFORNIA
13 SAN FRANCISCO DIVISION

14 IMPLICIT NETWORKS, INC.,

15 Plaintiff,

16 v.

17 JUNIPER NETWORKS, INC.,

18 Defendant.
19
20
21
22
23
24
25
26
27
28

Case No. C 10-4234 SI

**IMPLICIT NETWORKS, INC.'S
SUPPLEMENTAL RESPONSE TO
JUNIPER NETWORKS, INC.'S
SECOND SET OF
INTERROGATORIES (NOS. 11-19)**

1 Implicit was ordered to produce and identify any and all documents relevant to any partial
2 embodiments, including within that definition documents which discussed even one element
3 of what subsequently would constitute the patents-in-suit. Accordingly, Implicit identified
4 documents including any and all documents as so defined. These documents do not,
5 however, disclose all the element of any embodiments of the patents-in-suit, as those
6 embodiments did not exist until years later, as set forth above.
7

8 **Interrogatory No. 18:**

9 To the extent that Implicit contends that its asserted patent claims are not invalid for
10 the reasons set forth in Juniper's invalidity contentions and requests for reexamination for the
11 patents-in-suit, state the complete factual and legal basis for your contentions (including
12 identification of all supporting facts, documents, and persons with knowledge), including for
13 each prior art reference and combination of references disclosed by Juniper, a chart
14 identifying every claim element that you contend is not disclosed (either explicitly or
15 implicitly) in such prior art reference or combination of references, and your complete
16 alleged factual and legal basis and evidentiary support for such contention regarding each
17 such element.
18

19 **Response:**

20 Implicit refers to and incorporates by reference each of the foregoing General
21 Objections. In addition to the foregoing General Objections, Implicit specifically objects to
22 this interrogatory to the extent it is vague, ambiguous, overly broad, unduly burdensome, and
23 calls for information that is not relevant to the subject matter of this action or to a claim or
24 defense of any party and/or are not reasonably calculated to lead to the discovery of
25 admissible evidence. Plaintiff further objects to this interrogatory on the ground that it seeks
26
27
28

1 information protected by the attorney-client privilege, the work product doctrine, or any other
2 applicable privilege or protective doctrine.

3 This interrogatory is vastly overbroad. It asks that Implicit show, element by
4 element, why its claims are not invalid over the prior art references cited in any of the
5 Implicit cases by any of the defendants, along with all of the scores of prior art references in
6 Juniper's two inter partes re-exam petitions. It goes on to insist that Implicit provide validity
7 charts, claim by claim, along with all factual and legal bases and evidentiary support on each
8 subpoint.
9

10 Collectively, the defendants cited 480 individual pieces of prior art across their
11 multiple rounds of contentions. Many of the references are quite voluminous, and many are
12 quite general. The defendants cite, for example, general text books and lengthy manuals,
13 *e.g.:*
14

- 15 • Introduction to ATM Networking, Walter Goralski (395 pages)
- 16 • Principles of Information Systems Analysis and Design (444 pages)
- 17 • The Data Compression Book, 2nd Edition (574 pages)
- 18 • Digital Switching Systems ISDN Primary Rate User-Network Interface
19 Specification August, 1998 (545 pages)
- 20 • Software Engineering With ADA (577 pages)
- 21 • Asynchronous Transfer Mode (ATM) Technical Overview (2nd Edition) (325
22 pages)
- 23 • NetScreen Concepts & Examples ScreenOS Reference Guide 9336 pages)
- 24 • Local Area Network Concepts and Products: Routers and Gateways May 1996
25 (300 pages)
- 26 • JAVA Media Framework API Guide 11/19/99 (265 pages)

27 There any many such general references.

28 In addition, Juniper sets forth numerous additional references in its re-exam petitions.
In total, Juniper asks for charts plus all legal bases and all evidentiary support for over 500

1 separate references. Collectively, there are well over 6,000 pages in prior art references in
 2 the Invalidity Contentions and re-exam petitions.

3 Assuming just twenty pages of response per reference, which is likely low given all
 4 that Juniper demands, Interrogatory No. 18 requests an answer that would exceed 12,000
 5 pages. It would take multiple lawyers a year plus to answer this Interrogatory, as Juniper has
 6 framed it. Conceptually, it is as if Juniper simply cited Implicit to the Library of Congress,
 7 and asked that Implicit distinguish any and all references therein as not proving its claims
 8 invalid. Implicit met and conferred with Juniper on the scope of this Request and asked, *e.g.*,
 9 that Juniper limit the list of allegedly prior art references to those not *sua sponte* rejected by
 10 the PTO in the reexam responses. Juniper thus far has refused to reduce the scope of this
 11 Interrogatory.
 12

13 **SUPPLEMENTAL RESPONSE (7/25/12):**

14 For the six references Juniper has identified, Implicit responds as follows:

15 **Introduction**

16
 17 Implicit's '163 and '857 patents turn on creating a message-specific, stateful data
 18 processing path post-first packet. Using configuration information, including Label Map Get
 19 ("LMG"), the Implicit system "identifies," *i.e.* creates, a data processing path post-first
 20 packet to process the packets of that message, in a stateful manner. This was in
 21 contradistinction to prior systems, *e.g.* Mosberger, where the data processing paths were
 22 "predefined" in the sense of being fixed in the code and unalterable. A "non-predefined"
 23 message specific stateful data processing path is one actually created post-first packet, and as
 24 a consequence of receipt of the new message.⁴
 25

26 _____
 27 ⁴ Charts have been served for these six references as well. *See* Exhibits A-F.

None of the six references so operate.

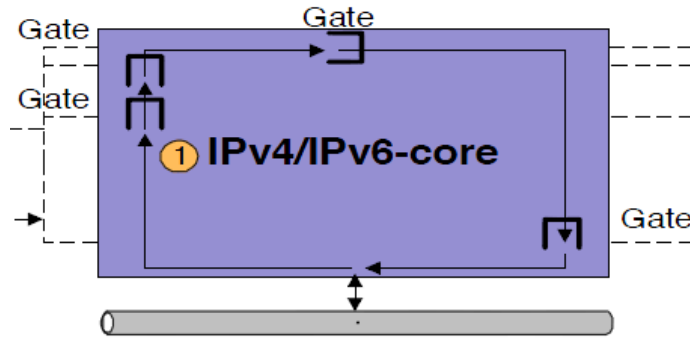
I. Decasper Fails to Render Claim 1 Obvious

A. Decasper Fails to Disclose “dynamically identifying a non-predefined sequence of components for processing the packets of the message..., wherein dynamically identifying includes selecting individual components to create the non-predefined sequence of components after the first packet is received”

Decasper describes a router architecture that consists of an IP router core containing a predefined sequence of numerous “components” and “gates” that process each packet received by the router. As explained below, the Decasper router only processes one data format (IP). There is no notion of changing data formats in this reference; it is not a demultiplexing system at all.

At the heart of Decasper’s router architecture lies an “IPv4/IPv6 core,” which “contains the ... components required for packet processing which do not come in the form of dynamically loadable modules.” Decasper at 3, col. 2 (emphasis added). Decasper expressly states, therefore, that the IP core “components” are not “dynamically” identified.

The core is also “responsible for demultiplexing individual packets to plugins” through a series of gates within the IP core. *Id.* When a packet arrives at the router, it is passed to the IP core by the network hardware. *Id.* at 5, col. 2. All packets are processed through the IP core, without exception. Figure 3 of Decasper, which depicts the overall system architecture and data path for an IP router, illustrates the processing path through the IP core via clock-wise arrows:



Excerpt from Figure 3 of Decasper

Each packet proceeds through the IP core in the same manner, until it encounters a gate. “A gate is a point in the IP core where the flow of execution branches off to an instance of a plugin.” Decasper at 4, col. 2. Gates are fixed at locations within the routing system where “interactions with plugins need to take place.” Decasper at 5, col. 1. Decasper denotes gates in Figure 3 by an “II” symbol within the IP core. *See* Figure 3 excerpt, above. Plugins are “bound” to a gate during the configuration of the router. Decasper at 4, col 1. The “task” of a gate is to determine if a packet (based only on its header) needs to be processed by a plugin bound to the gate and—if so—which one. Decasper at 5, col 2. These plugins represent extensions to the IP core of the router that provide optional functions such as IP security and packet scheduling. Decasper at 6, col 2. As shown, the location of the gates within the IP core processing path is fixed, and the location of optional plug-in function calls is also fixed (since plug-ins are bound to the gates prior to the receipt of any packets).

It is clear, therefore, that all packets in Decasper receive the same general processing through the IP core, and all packets are processed by the same predetermined number of gates in the same predetermined order. The only difference between the processing paths of packets is whether or not they branch out at a specific location (a gate) to a plugin.

The above discussion makes clear that the sequence of components and gates within the Decasper IP core is fixed before the first packet arrives. As such, Decasper does not

1 “dynamically identif[y] a non-predefined sequence of components for processing the packets
 2 ..., wherein dynamically identifying includes selecting individual components to create the
 3 non-predefined sequence of components after the first packet is received.”

4 **B. Decasper Fails to Disclose “dynamically identifying a non-predefined
 5 sequence of components for processing the packets of the message such
 6 that the output format of the components of the non-predefined sequence
 7 match the input format of the next component in the non-predefined
 8 sequence”**

9 Decasper does not disclose processing the packets of the message **such that** the
 10 output format of the components match the input format of the next component.

11 Decasper discloses an architecture for an IP routing system. An IP router and its
 12 components, by definition, are only designed to handle packets in the IP (Internet Protocol)
 13 format—a single format; this is a one format system. Accordingly, the Decasper architecture
 14 presupposes that all packets will be in IP format, and therefore that the input and output of
 15 every component will be the same format: IP. Given the single format of data (IP) that
 16 Decasper processes, it teaches away from considering **format compatibility** when
 17 “dynamically identifying a non-predefined sequence of components.”

18 Nor would there be any reason to consider format compatibility in Decasper. The
 19 architecture is specifically designed to handle one format (IP), and thus matching different
 20 formats is extrinsic to the reference. **In fact, adding functionality to check or compare the**
 21 **output format of a packet processed by one IP component with the input format required by**
 22 **the next IP component would add unnecessary and costly overhead. In short, no skilled**
 23 **person looking at Decasper would add or consider this claim requirement.**

24 **C. Decasper Fails to Disclose the State Information Limitations**

25 Claim 1 requires “for each of a plurality of components in the identified non-
 26 predefined sequence, [1] retrieving state information relating to performing the processing of
 27

1 the component with the previous packet of the message; [2] performing the processing of the
2 identified component with the packet and the retrieved state information; and [3] storing state
3 information relating to the processing of the component with packet for use when processing
4 the next packet of the message.” The Court has construed state information as “**information**
5 **specific to a software routine for a specific message** that is not information related to an
6 overall path.” *Markman* Order at 14 (Ex. 2) (emphasis added). Decasper discloses none of
7 the claimed requirements of “retrieving,” “processing” and “storing” message-specific state
8 information for the components.
9

10 Conventional IP routers from the 1990s—like Decasper and Kerr—do not
11 contemplate the claimed state information. This is because all of the information needed to
12 route a given IP packet is contained within the IP packet per the specification for the Internet
13 Protocol. These conventional IP routers operating at the packet level do not require state
14 information created by the processing of one packet to process a subsequent packet. Such IP
15 routers simply do not contemplate this type of state management.
16

17 Decasper does maintain a flow index (FIX), which is a “pointer to the row in the flow
18 table where the information associated with the flow is stored,” or the “instance pointers”
19 stored within the flow table, which point “to the appropriate plugins for all gates that can be
20 encountered by packets belonging to the corresponding flow.” For the reasons discussed
21 below, these pointers do not constitute “information specific to a software routine” that
22 “relate[s] to performing of the processing of the component” with a packet or any other state
23 information limitation, as required by claim 1.
24

25 Decasper makes it clear that the FIX pointer “is stored in the packet’s mbuf” and that
26 “instance pointers cached in the flow table” are retrieved “by accessing the FIX stored in the
27 packet.” *Id.* These pointers are not “information specific to a software routine”—the FIX is
28

1 stored in a packet's buffer, and is *therefore the same for each and every plugin*. The instance
2 pointers in a flow table are also fixed on a per-packet or per-message basis (rather than a per-
3 component) basis, and point to plug-ins that should process packets of a particular flow.

4 Accordingly, they do not constitute "state information," as this term is used in the claims.

5 Furthermore, neither the FIX pointer stored in a packet's "mbuf" nor the instance
6 pointers stored in the flow table are retrieved, used, or stored by plugins (and therefore they
7 are not used to process packets), as required by claim 1. The FIX pointer is used by gates
8 within the IP core to retrieve the instance pointers cached in the flow table. Decasper at 6, col
9 1. Once the gate receives the instance pointer for a particular packet, it passes the packet to
10 the plugin pointed to by the instance pointer. Decasper at 6, col 1; 5, col. 2. The plugin then
11 processes the packet, and returns the packet to the IP core. Decasper at 5, col 2. In other
12 words, the sole purpose or function of these pointers is to tell gates where to send a packet
13 for processing—either to a plugin or along the IP core. These pointers are not accessed or
14 utilized by the plugins in any way, and they are certainly not used as the claimed state
15 information. There is no discussion in Decasper about information related to the processing
16 of packets being used by plugins to process the packets, and then stored for use in the
17 processing of a later packet. *The plugins of Decasper do not need to know anything about*
18 *how the previous packet of a message was processed—they simply receive a packet from the*
19 *IP core, process it, and return it to the IP core.* Decasper at 5, Column 2. Accordingly, the
20 use and storage of state information as claimed is not disclosed in Decasper.
21
22
23

24 Nor do these pointers contain "information relating to the performing of the
25 processing of the component." The instance pointers store the location of the particular
26 plugins to which a gate should send packets associated with a particular flow, and the FIX
27 pointers store the location of the row of the flow table that contains the instance pointers for a
28

1 particular flow. This “addressing” information does not constitute the claimed “state
2 information.”

3 For the above reasons, Decasper neither teaches nor discloses the claimed state
4 information.

5 **B. Decasper Fails to Anticipate Claims 15 and 35**

6 **1. Decasper Fails to Disclose the Requirements in Claims 15 and 35**
7 **Regarding Dynamically Identifying a Non-Predefined Sequence of**
8 **Components**

9 As with claim 1, claims 15 and 35 require “dynamically identifying” a “non-
10 predefined sequence of components.” The Court construed “non-predefined sequence of
11 components” as “a sequence of software routines that was not identified before the first
12 packet of a message was received.” *Markman* Order at 6 (Ex. 2). For the same reasons set
13 forth above, Decasper also fails to disclose these requirements in claims 15 and 35.

14 **2. Decasper Fails to Disclose the Requirements of Claims 15 and 35**
15 **Regarding “selecting individual components to create the**
16 **[message-specific] non-predefined sequence of components”**

17 The Court construed “selecting individual components,” which appears in claims 15
18 and 35, as “selecting the individual software routines of the sequence so that the input and
19 output formats of the software routines are compatible.” *Markman* Order at 11 (Ex. 2).

20 As explained above, Decasper does not disclose the required input/output format
21 compatibility. Nor would such compatibility be obvious.

22 **3. Decasper Fails to Disclose the Requirements of Claims 15 and 35**
23 **Regarding “State Information”**

24 Claim 15 requires that “for each packet of each message, performing the processing
25 of the identified non-predefined sequence of components of the message wherein state
26 information generated by performing the processing of a component for a packet is available
27

1 to the component when the component processes the next packet of the message.” Claim 35
 2 requires that “for each packet of each message, invoking the identified non-predefined
 3 sequence of components in sequence to perform the processing of each component for the
 4 packet wherein each component saves message-specific state information so that that
 5 component can use the saved message-specific state information when that component
 6 performs its processing on the next packet of the message.” For the same reasons set forth
 7 above, Decasper also fails to disclose these state information requirements in claims 15 and
 8 35.
 9

10 II. Kerr Neither Anticipates Nor Renders the Patents-in-Suit Obvious

11 A. Kerr Fails to Disclose “dynamically identifying a non-predefined 12 sequence of components for processing the packets of the message..., 13 wherein dynamically identifying includes selecting individual components 14 to create the non-predefined sequence of components after the first 15 packet is received”

16 Kerr does not disclose “dynamically identifying a non-predefined sequence of
 17 components” for processing the packets of a message. In fact, Kerr does not disclose
 18 identifying components at all, much less the dynamic identification of a “non-predefined
 19 sequence of components” for processing packets. Instead, Kerr discloses a routing device that
 20 “determines the proper processing” for packets and then stores information regarding that
 21 processing:

22 When routers in a network identify a new message flow, they determine the
 23 proper processing for packets in that message flow and cache that
 24 information for that message flow. Thereafter, when routers [receive a
 subsequent packet of the message] they process that packet according to the
proper processing for packets in that message flow.

25 Kerr at 1:52-55 (emphasis added); *see also* Abstract.

26 The routing device 140 determines the proper treatment of packets 150 in
 27 the message flow 160 and enters information regarding such proper
 28 treatment in a data structure pointed to by the new entry in the flow cache.

1 Kerr at 4:12-20 (emphasis added). Accordingly, Kerr does not disclose “selecting individual
2 components” for processing packets of a message as required by claim 1, but merely
3 discloses the determination of the correct processing for the packets of a message, and the
4 storage of information regarding that processing.
5

6 At column 4, lines 20-34, it discusses what the “**proper treatments of packets** ... in
7 the message flow” might be, with regards to “switching,” “access control,” “accounting,” or
8 “any special treatment for packets.” Importantly, this passage—like all of the other passages
9 in the reference—is discussing the determination of the **treatment** or **processing** of packets.
10 Kerr never discusses or discloses the identification or selection of software routines for
11 performing said processing or treatments, nor the “dynamic identification” of a “non-
12 predefined sequence” of such components. In fact, Kerr does not discuss any sort of
13 software routines.
14

15 Furthermore, Kerr makes clear that it is the information about the complete proper
16 treatment or processing of packets that is stored in the flow cache—not an identification of a
17 particular software component. “The flow cache 300 ... includes information about a
18 particular message flow 160, including ...access control, accounting, special treatment for
19 packets 150 in that particular message flow, and a pointer to information about treatment of
20 packets 150.” Kerr at 6:32-42. When a routing device receives a subsequent packet in a
21 flow, “[it] reads the information from the entry in the flow cache and treats the packet 150
22 according to the information in the entry in the flow cache.” Kerr at 4:67-5:4.
23

24 Nothing in the other sections of Kerr discloses anything other than the determination
25 of proper processing and the storage of information about that processing; in Kerr, the
26
27
28

1 routing device “determines proper treatment of packets” and “enters information regarding
2 such proper treatment” in a data structure.

3 Furthermore, in the on-going reexamination of the continuation patent based on the
4 ’163 patent, ’857 reexam (U.S. Pat. No. 7,711,857; Reexamination Control No. 95/000,660),
5 Examiner Kenneth J. Whittington stated in his first Office Action that Kerr does not teach the
6 identification of a sequence of components for processing the packets of a message, much
7 less dynamic identification:
8

9 Kerr does not explicitly teach any identification of a sequence of components
as part of its proper treatment of a message flow.

10 ’857 5/10/2012 Office Action at 17, 22, 26. More:

11 As noted above ... proper treatment within [Kerr’s] router architecture
12 comprises multiple processing stages ... but [Kerr] does not specifically
13 outline the detail or structure thereof.

14 ’857 5/10/2012 Office Action at 20, 23, 27.

15 For the above reasons, Kerr fails to disclose “dynamically identifying a non-
16 predefined sequence of components” for processing the packets of a message and so does not
17 anticipate.

18 **B. Kerr Fails to Disclose “dynamically identifying a non-predefined
19 sequence of components for processing the packets of the message such
20 that the output format of the components of the non-predefined sequence
21 match the input format of the next component in the non-predefined
sequence””**

22 Kerr does not disclose “processing the packets of the message such that the output
23 format of the components match the input format of the next component.”

24 Kerr discloses a system and method for packet handling within an IP router. Kerr
25 never discloses or discusses the format of the data packets processed by its system.

26 Presumably, this is because Kerr limits its disclosure to an IP router and its components,
27
28

1 which, by definition, are only designed to handle packets in the IP (Internet Protocol)
 2 format—a single format. Nor is there any evidence that the “treatments” in Kerr change the
 3 format of the packet. Accordingly, Kerr assumes that all packets will be in IP format, and
 4 therefore that the input and output of every component will be a packet with an IP format.
 5 Given the unitary format of data (IP) that Kerr processes, it teaches away from **considering**
 6 **format compatibility when “dynamically identifying a non-predefined sequence of**
 7 **components.”**

9 Nor would there be any reason to consider format compatibility in Kerr. Kerr is
 10 designed to handle one format (IP), and one of ordinary skill would be unconcerned about
 11 matching different formats. **In fact, adding functionality to check or compare the output**
 12 **format of a packet processed by one IP component with the input format required by the next**
 13 **IP component, would add unnecessary and costly overhead, as noted above. In short, no**
 14 **skilled person looking at Kerr would add the claim requirement.**

16 Furthermore, as explained above, Kerr does not discuss any “components” or
 17 “sequence of components”—its disclosure is limited to the determination of the proper
 18 treatment or processing of packets. Thus, there is also no disclosure of dynamically
 19 identifying a non-predefined sequence of components for processing the packets of the
 20 message “such that the **output format of the components** of the non-predefined sequence
 21 **match the input format of the next component** in the non-predefined sequence.”

23 In sum, Kerr does not disclose and indeed teaches away from the input/output
 24 compatibility requirement.

25 C. Kerr Fails to Disclose the “State Information” Limitations

26 Claim 1 requires “for each of a plurality of components in the identified non-
 27 predefined sequence, [1] retrieving state information relating to performing the processing of
 28

1 the component with the previous packet of the message; [2] performing the processing of the
2 identified component with the packet and the retrieved state information; and [3] storing state
3 information relating to the processing of the component with packet for use when processing
4 the next packet of the message.” The Court construed state information as “**information**
5 **specific to a software routine for a specific message** that is not information related to an
6 overall path.” *Markman* Order at 14 (Ex. 2) (emphasis added). Kerr discloses none of the
7 claimed requirements of “retrieving,” “processing” and “storing” message-specific state
8 information for the components.
9

10 Conventional IP routers from the 1990s—like Decasper and Kerr—by their very
11 nature do not require packet-to-packet state information since each IP packet can be and is
12 routed independently. It is not surprising, therefore, that Kerr never mentions “state” or
13 “state information.”
14

15 Additionally, for the reasons discussed above, Kerr does not disclose the
16 identification of a “sequence of components.” Instead, it teaches that routing devices
17 “determine[] proper treatment of packets” and “enter[] information regarding such proper
18 treatment in a data structure.” Accordingly, Kerr fails to disclose retrieving, using, or storing
19 of “information specific to a software routine for a specific message” “relating to performing
20 the processing of the component,” as required by claim 1.
21

22 For the above reasons, Kerr does not disclose the claimed state information. In fact,
23 Kerr does not even mention state or state information. For these and similar reasons, the
24 PTO found that Kerr was not an anticipatory reference. *See* below, § IV, “**Pfeifer Does Not**
25 **Anticipate.**”

26 **D. Kerr Fails to Anticipate Claims 15 and 35**
27
28

1 1. **Kerr Fails to Disclose the Requirements in Claims 15 and 35**
 2 **Regarding Dynamically Identifying a Non-Predefined Sequence of**
 3 **Components**

4 Like claim 1, claims 15 and 35 require “dynamically identifying” a “non-predefined
 5 sequence of components.” The District Court construed “non-predefined sequence of
 6 components” as “a sequence of software routines that was not identified before the first
 7 packet of a message was received.” *Markman* Order at 6 (Ex. 2). For the same reasons set
 8 forth above with respect to claim 1, Kerr also fails to disclose these requirements in claims
 9 15 and 35.

10 2. **Kerr Fails to Disclose the Requirements of Claims 15 and 35**
 11 **Regarding “selecting individual components to create the**
 12 **[message-specific] non-predefined sequence of components”**

13 The District Court construed “selecting individual components,” which appears in
 14 claims 15 and 35, as “selecting the individual software routines of the sequence so that the
 15 input and output formats of the software routines are compatible.” *Markman* Order at
 16 11 (Ex. 2). As explained above, the PTO agrees that Kerr does not disclose the required
 17 input/output format compatibility. Nor would such compatibility be obvious for the same
 18 reasons set forth above.

19 3. **Kerr Fails to Disclose the Requirements of Claims 15 and 35**
 20 **Regarding “State Information”**

21 Claim 15 requires that “for each packet of each message, performing the processing
 22 of the identified non-predefined sequence of components of the message wherein state
 23 information generated by performing the processing of a component for a packet is available
 24 to the component when the component processes the next packet of the message.” Claim 35
 25 requires that “for each packet of each message, invoking the identified non-predefined
 26 sequence of components in sequence to perform the processing of each component for the
 27
 28

1 packet wherein each component saves message-specific state information so that that
 2 component can use the saved message-specific state information when that component
 3 performs its processing on the next packet of the message.” For the reasons set forth above,
 4 Kerr also fails to disclose these state information requirements in claims 15 and 35.

5 **III. Mosberger’s Scout Neither Anticipates Nor Renders the Patents-in-Suit Obvious**

6 **A. Mosberger’s Predefined Paths**

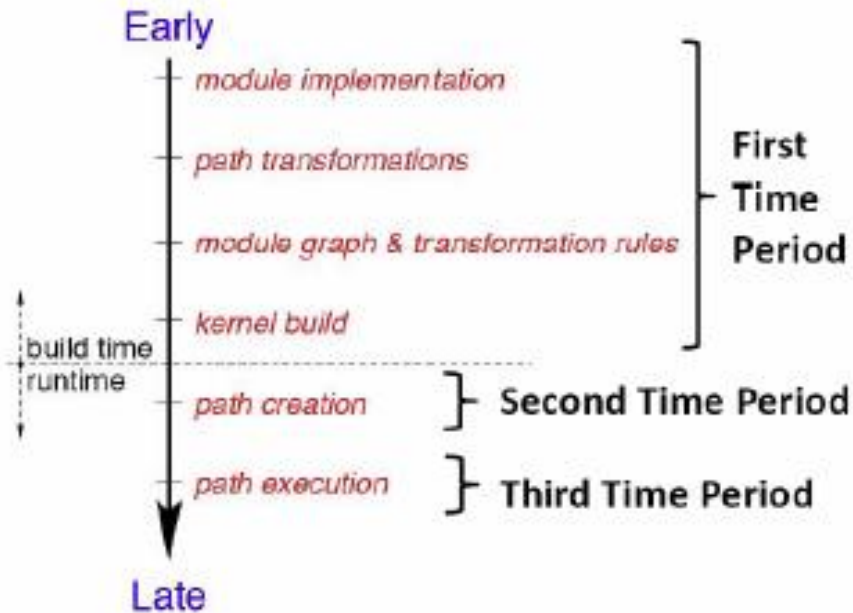
7 Mosberger’s “Scout” was a processing system where code processing components
 8 were compiled together at build time to create a fixed and invariant set of processing paths.
 9 The paths existed pre-first packet, and indeed came into being when the system was powered
 10 up but not yet connected to a network.⁵ The system was designed to work with purpose built
 11 “appliances,” *e.g.*, a relatively finite number of clients processing data in predictable ways.
 12

13 There are three key “epochs” or time periods during the development and operation
 14 of the Scout path-based operating system. (Mosberger, pp. 60-61).⁶ The first is “build time,”
 15 when the programmer designs the individual modules, decides what kinds of paths are likely
 16 to be important to system performance, develops the module graph and builds the system
 17 kernel. *Id.* The second time period is “path creation,” which occurs at “runtime” during
 18 system initialization when the system creates all the possible paths. (*Id.*; Mosberger, pp. 80-
 19 82). The third and final time period is “path execution,” which also occurs at “runtime” (but
 20 after “path creation”) when the system receives messages, selects (or finds) which of the
 21 predefined paths is appropriate for a particular message and then executes the modules in that
 22 predefined path. (Mosberger, pp. 60-61, 85, 100-101). These three time periods are
 23
 24

25 ⁵ Once a path was in use, the system would create a mirror path to await the next flow. This was just a
 26 copy of the pre-existing, pre-defined path.

27 ⁶ All references are to Mosberger’s “Scout” doctoral dissertation.

illustrated in the following annotated figure from page 61 of Mosberger (the annotations include the brackets and language to the far right which are added for emphasis):



Section 3.3 of Mosberger describes how the paths in Scout are “realized.” (Mosberger, pp. 71-85). The first part of Section 3.3 explains the basic components of a path (*i.e.*, “modules,” “stages” and “interfaces”). (Mosberger, pp. 71-80). A “module” is a “unit of program development in Scout” that “provides a well-defined and independent functionality.” (Mosberger, pp. 61-62). And “there is one stage per module that the path traverses.” (Mosberger, p. 73). Finally, “an interface provides a controlled (type-checked) way to move data from one stage to the next one.” (Mosberger, p. 75). Figure 3.5 of Mosberger illustrates the relationship between the various components in the path:

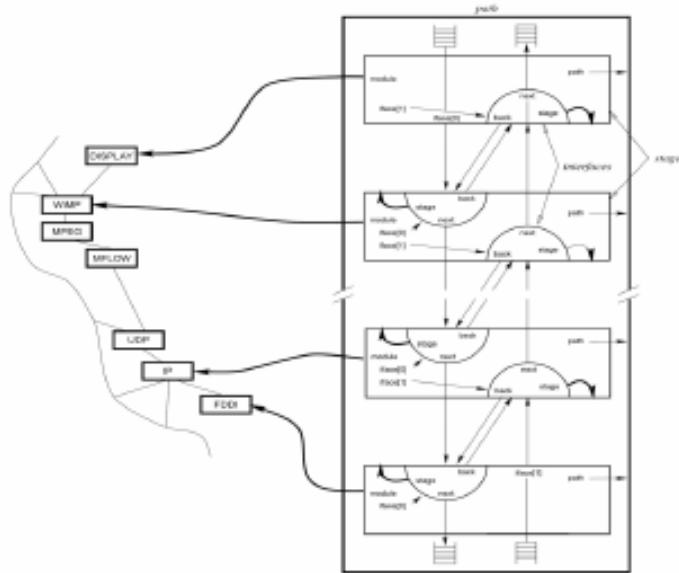


Figure 3.5: Path Structure

The next part of Section 3.3 (titled “3.3.6 Creation”) describes how paths are created during the “path creation” time period. (Mosberger, pp. 80-82). Mosberger teaches that the “pathCreate” software function actually creates the paths. (Mosberger, p. 80). The C-programming language prototype of the “pathCreate” software function is:

```
Path pathCreate (Module m, Attrs a),
```

(*Id.*). As its prototype suggests, a path is created by invoking the “pathCreate” software function does not take a message as a parameter, showing that the paths are created independently of the messages. (*Id.*). This is not surprising, considering that the paths are created during initialization – before any messages/packets can be received.

The “pathCreate” software function eventually invokes the “createStage” software function, which has the following prototype:

```
Stage (*CreateStageFunc) (Module m, int s, Attrs a, ModuleLink* n);
```

(*Id.*). This prototype shows that the parameters for the “createStage” software function are a module “m,” a service index “s,” a set of attributes “a,” and a pointer “ModuleLink*” to the service index of the next stage in the path. (*Id.*). Like the “pathCreate” software function,

1 the “createStage” function also does not have an input parameter for a message, which
2 further shows that the stages are created without regard to the particular messages. (*Id.*).

3 The end result of invoking the “pathCreate” software function is that Scout will
4 create a set of paths comprising various sequences of modules. (Mosberger, p. 81: “At this
5 point, the pathCreate function creates the actual path object, inserts the stages into it, and
6 establishes the various chains through the path structure”). Importantly, this set of paths is
7 finite; Mosberger does not teach to create new paths after initialization. (*Id.*). Further, this
8 set of paths is created before any message/packet is received. (*Id.*).

10 The next section of Mosberger, Section 3.4 entitled “Demultiplexing,” describes how
11 to select the appropriate path from amongst the finite set of previously-created paths based on
12 the contents of a particular message. (Mosberger, pp. 85-99). This point is underscored by
13 the first sentence of section 3.4: “**So far**, we have not discussed the issue of how the
14 appropriate path **is found** for a given message.” (Mosberger, p. 85) (emphasis added). This
15 sentence reflects that the prior sections of Mosberger regarding creating paths are limited to
16 “path creation” and do not relate to selecting (or “finding”) the appropriate (predefined) path
17 for a particular message. (*Id.*). Instead, section 3.4 reflects that, in Mosberger, upon the
18 receipt of a message, Scout uses a demultiplexing process to **find** the correct previously-
19 created path to process the message. (Mosberger, pp. 85-92). Thus, paths in Mosberger are
20 not dynamically created based on the receipt of a message. (*Id.*). Rather, Mosberger teaches
21 that when a message is received, a path is selected (or “found”) from a set of possible paths,
22 which were created and “predefined” before the message was even received.

25 It was to distinguish this type of predefined path that Implicit added the “dynamically
26 identify” and “non-predefined” language in the ’163 re-exam, as set forth in the claims
27 briefing.

1 Third party scientists have recognized that '163 is profoundly unlike the Scout
2 system. For example, Dr. Donnelly of the University of Cambridge, subsequently of
3 Microsoft, described this point as follows:

4 **Scout is a static system which is configured and type-checked**
5 **at build-time. It does not support the dynamic loading of new**
6 **code modules. Its typesystem is fixed, and new types cannot be**
7 **added at run-time. This is unsurprising, given its original**
8 **niche as an OS for Internet appliances.**

9 Despite these problems with Scout, the concept of using paths as
10 schedulable entities is sound: per-path resource control directly
11 simplifies the resource allocation problem by providing one
12 parameter per resource that governs all processing applied to a
13 particular flow of packets.

14 ***

15 More recently, InfoPipes [Koster01] and **Strings of Beads**
16 **[BeComm]** are both component frameworks where data travels
17 along paths between code modules with strongly typed interfaces.
18 This allows the middleware to type-check module chains, thus
19 ensuring correct data flow. **The Strings framework goes further,**
20 **using a prolog-like unification engine to infer the correct**
21 **sequence of code modules to compose in order to meet goal**
22 **conditions set in a system configuration file.**

23 “Resource Control in Network Elements,” Austin Donnelly, April 2002, at 14, 38 (emphasis
24 added).

25 **A. Mosberger Does Not Dynamically Create Message-Specific and Stateful**
26 **Data Processing Paths**

27 Claims 1-25 and 35-44 each recite the step of “dynamically identifying a non-
28 predefined sequence of components” and further explain that “dynamically identifying
includes selecting individual components to create the non-predefined sequence of
components after the first packet is received” or variations thereof. (Emphasis added).

The amended claims are clear, therefore, that: (i) the identification of the sequence is
“dynamic”; (ii) what is identified is a “non-predefined” sequence; (iii) the dynamic

1 identification occurs after the first packet is received or based on the first packet; and (iv) the
2 dynamic identification includes creating the non-predefined sequence after the first packet is
3 received or based on the first packet. As discussed above, Mosberger does not so operate.
4 Mosberger does not dynamically identify a non-predefined sequence of components, which
5 includes creating the non-predefined sequence after the first packet is received or based on
6 the first packet. Instead, Mosberger creates the paths containing the sequence of modules
7 **before** the first packet is received. (Mosberger, pp. 61, 71-95). Thus, by the time the first
8 packet is received in Mosberger, the paths have been created and the system merely “finds”
9 the appropriate predefined path(s).
10

11 For these reasons, the PTO rejected Mosberger as an anticipatory reference in the
12 ’163 (re) re-exam, as discussed further below.

13 **IV. Direct Show Neither Anticipates Nor Renders the Patents-in-Suit Obvious**

14 Microsoft’s “Direct Show” was a successor to Microsoft’s ActiveMovie. The first
15 iteration of DirectShow was released circa July 1997. Its functionality changed over
16 subsequent years.
17

18 Microsoft’s Direct Show turned on processing steps called “filters.” Each filter had
19 an input or output “pin” that was used to connect a given filter to other filters. These pins
20 enabled filters to be connected together to accomplish different processing functionalities.
21 To implement a specific processing function, the developer had to build initially a “filter
22 graph”⁷ by connecting all of the requisite filters together. Processing is driven by file type,
23 *e.g.*, AU1 or JPEG; it is not a networking demux system at all.
24
25

26
27 ⁷ Much like the Mosberger “router graph.”

1 In Direct Show, there was no real-time mapping of incoming packets to flows, nor
 2 provision for packet inspection to assist in filter mapping. There was no provision for a state
 3 managed on behalf of the filters, as each filter had to manage its own state. Nor was there
 4 any support for basic networking processes, *e.g.*, Ethernet, TCP, IP, UDP, and the like.

5 Importantly, Microsoft's Direct Show was a **media processing system**: it was
 6 designed to process media and media alone. It was not a general network processing engine.

7 **V. Pfeifer Neither Anticipates Nor Renders the Patents-in-Suit Obvious**

8 Pfeifer, "Generic Conversion of Communication Media for Supporting Personal
 9 Mobility," is a media conversion system. It turns on the following concepts:
 10

- 11 • Converters reside in a system to convert data from one format, or medium, to
 12 another.
- 13 • A message is received and stored.
- 14 • The system determines where a user is located and based on that location what
 15 type of terminal and terminal format to receive the messages (based on user
 16 set rules from the iPCSS).
- 17 • Based on user location and terminal requirements, message is converted into
 18 appropriate format using different conversion components chained together.

18 Specific functional details are as follows:

- 19 1. Data is received by the Service Gateway (Pfeifer 5.2.3.1)
 20 a. Data can be telephony, email, fax or Internet
- 21 2. The Service Gateway processes incoming messages in the following ways:
 22 a. Determine the called party (via transmitted communication address)
 23 b. Determine the calling party (via transmitted address or other)
 24 c. Describe Service by filling out a Teleservice Descriptor with parameters
 25 d. Map calling/called party to internal UID of the iPCSS system e. Store
 26 Message in Active Store
- 27 3. Active Store invokes the intelligent Communication Manager (iCM).

- 1 4. The iCM is responsible for controlling all the system behavior.
2 Responsible for the of a single communication request. iCM is a state
3 machine.
- 4 5. User Related Data:
 - 5 a. Personal Call Logic (*PCL*) - information about the settings of the
6 communication environment of each iPCSS user. Provides the rules and
7 actions calls. Rules can be edited by user using Call Logic Editor.
 - 8 b. Terminal Registration (*TReg*) - information about the actual manual
9 terminal of iPCSS users.
 - 10 c. Personal Schedule (*PSch*) - information about pre-planned
11 regular locations or absences.
 - 12 d. User Location (*ULoc*) - current or default location of iPCSS users.
- 13 6. CPE Selection and Config –
 - 14 a. Terminal Agent (TA) - Object that stores information about the
15 users system(s). Maintains capabilities of a specific terminal and used by
16 the Basic Resource Selector.
 - 17 b. Mediator Agent (MA) - Object that contains description of
18 converters.
- 19 7. Basic Resource Selector (BReS) - maintains information about
20 communication capabilities at locations.
- 21 8. Intelligent Resource Selector (IReS) - Service Supporting Object for the
22 BReS that maintains knowledge about a pool of terminals/applications (TA's)
23 and converters (MA). The IReS dynamically calculates the most appropriate
24 converter chain depending on the requested Teleservice.
- 25 9. Resource Configurator (ReCo) - is a Service Supporting Object for the
26 IReS. Is has the ability to configure converter chains and to control the stream
27 binding of them. The abstract MA/TA connection graph (including converter
28 chain) is delivered to ReCo. ReCo control config tasks via the Converter
 Framework (CF).
10. Yellow Pages - maps communication addresses (phone numbers, email
 accounts, etc.) to iPCSS User ID's.

A. The PTO Rejected Pfeifer as a Reference

The PTO has already rejected Pfeifer has an anticipatory reference, either alone or in
conjunction with Kerr and Mosberger. To quote the Office:

Pfeifer96 teaches the Intelligent Personal Communication Support System is introduced as an application for multiple media conversion tools, embedded in a context of personal mobility, service personalization and service interoperability support.

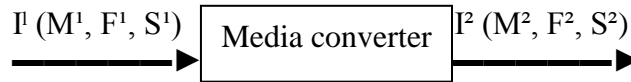


Fig. 1 Media converter system

After discussing models for conversion in theory, the current conversion technology is evaluated. The necessity of an integrated framework of flexible converters and a generic converter model are derived and automatic management of conversion quality is discussed (Abstract).

Pfeifer96 further teaches

For converting one format of the same medium into another, tools exist for many platforms for text, bitmap images and audio. They are mostly in the public domain, and perform well as software solutions [32]. Converting video formats requires the appropriate encoding/decoding hardware and software for the compression methods involved [28, 29, 30, 31].

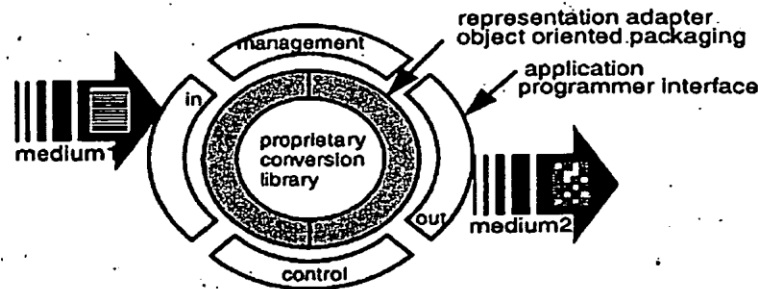


Fig. 7. Generic converter model

Finally, Pfeifer96 teaches:

In order to illustrate the benefits of the PCS concept, Figure 10 depicts a simplified intelligent call processing model to be performed by an advanced Personal Communication platform. It is characterized by a *four-stage mapping process* that translates a logical user name used as the called party address (i.e. a personal ID) into an appropriate network address (i.e. a terminal ID). This temporary physical address is passed back to the requesting communication service. The mapping process looks as follows:

- *1st*, the evaluation of a user's "Personal Call Logic" provides the *control of his reachability*. The result may be a forwarding to another user, a call rejection, a call redirection to an asynchronous service, e.g. an answering machine, or an acceptance.
- *2nd*, the exact recipient of the communication invitation has been settled and no further call management will be performed. A *mapping of the user to his location* is made based on user registration data.

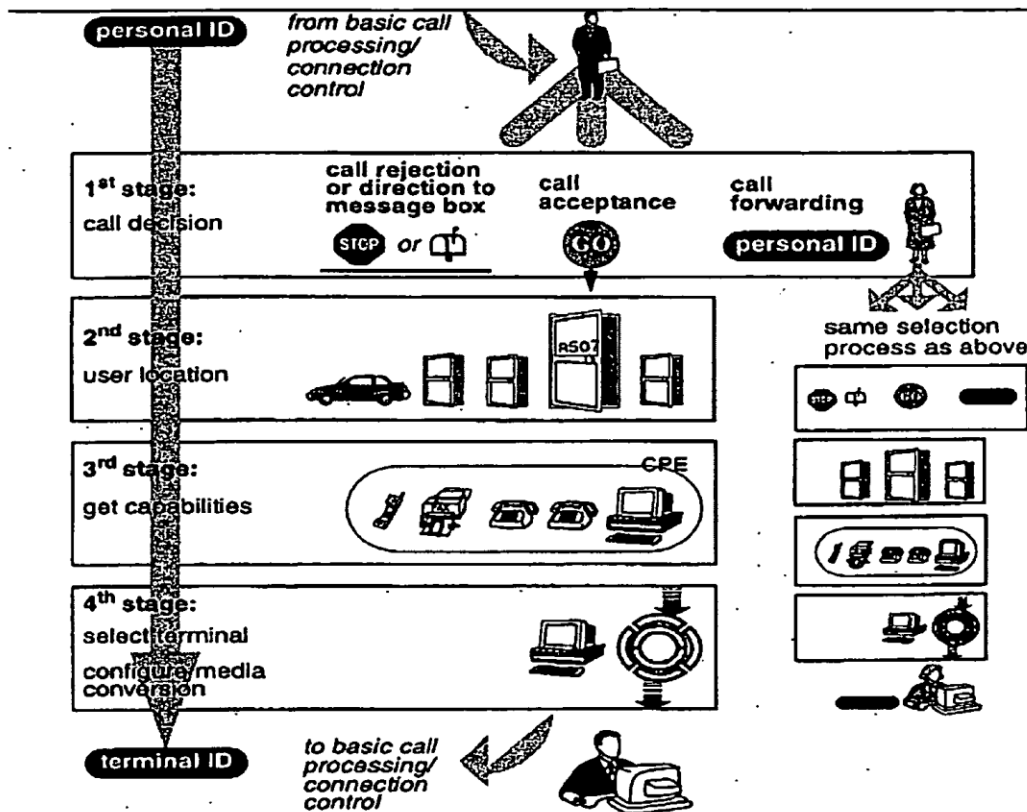


Fig. 10. PCS-based Intelligent Call Processing

- **3rd**, it maps a location to a virtual communication endpoint corresponding to a terminal group representing the set of all access devices in the user's current vicinity. An object-oriented modelling of virtual communication endpoints encompasses the knowledge on terminal capabilities; supported services, and selection mechanisms.
- **4th**, an appropriate terminal ID from the group of devices is selected and parameterized by a service type, used communication media, and optionally by user preferences. Within this stage, two cases can be distinguished:
 - a) In case there exist at least one device of the virtual communication end-point supporting the desired medium of the call, the most appropriate device is selected.
 - b) In case no device for the desired medium can be found, further rules of the Personal Call Logic determine whether a *conversion into another medium* is allowed/restricted. Then, the necessary converters are configured and a now appropriate device is selected.

The PTO found, correctly, that Pfeifer does **not** turn on a first packet identifying a series of components, much less dynamically identifying such a series. Nor does the reference retrieve or store state information. For these reasons, the PTO has previously found that Pfeifer does not anticipate nor render the patents-in-suit obvious. This is quite right.

VI. HotLava Neither Anticipates Nor Renders the Patents-in-Suit Obvious

HotLava, “Implementing Protocols in Java: The Price of Portability,”⁸ discusses various aspects of Java-based protocol subsystem design. The PTO has already rejected HotLava as an anticipatory reference, either alone or in connection with Mosberger’s Scout. To quote the Office:

For the **first packet** of the message, **identifying** a sequence of components for processing the packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence; and **storing** an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message...

Similarly, claim 15 states:

...**identifying** a sequence of components for processing each message based on the **first packet** of the message so that subsequent packets of the message can be processed without re-identifying the components,...

Similarly, claim 35 states:

...**identifying** a message-specific sequence of components for processing the packets of each message upon receiving the **first packet** of the message wherein subsequent packets of the message can use the message-specific sequence identified when the **first packet** was received...

In regards to these limitations relating to “**first packet**,” Requester submits in pages 262-264:

HotLava expressly describes itself as a “dynamic” system: “In our Java-based protocol architecture, special service classes dynamically construct protocol graphs at runtime as applications need communications services.” *Id.* at 96; *see also id.* at Fig. 1.

HotLava explains that it is “natural to consider” the Java environment as a way of addressing the need for “flexible

⁸ Bobby Krupczak, Kenneth L. Calvert, Mostafa Ammar, College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280 {rdk,calvert,ammar}@cc.gatech.edu.

1 communication protocols and services to support them,” as a way
 2 of solving the problem of “the number and variety of Web – and
 3 network-based applications [that] continue[] to increase.” *Id.* at
 4 93. Using the system disclosed in HotLava, “protocols and
 5 additional code required to support them can be downloaded and
 6 executed on the fly as needed.” *Id.* See also *id.* at 96 (“This
 7 extensible architecture..., allows on-the-fly introduction of new or
 8 replacement protocol code.”). Thus, “new classes, such as those
 9 making up our protocol subsystem and protocol implementations,
 10 can be added dynamically....” *Id.* at 95. Among other things, the
 11 HotLava approach overcomes shortcomings of certain “traditional”
 approaches and systems, which had to be “completely recompiled
 and redeployed” in order to accommodate change. *Id.* Not only is
 the protocol graph of software modules (“sequence of protocols”)
 determined “dynamically... at runtime,” each also receives a
 separate instantiation in memory; thus, “multiple instances of the
 same protocol can be executing simultaneously.” *Id.* at 98. “For
 example, an application needing AppleTalk services need only
 create an instance of its corresponding service class.” *Id.* at 96.

12 As explained earlier, Mosberger proposed configuring “a virtual
 13 machine module into the graph that would allow interpreted code
 14 to be downloaded and executed inside Scout.” Ex. 31 at 71. As
 15 shown above, HotLava expressly provides “the ability to
 16 incorporate new protocol classes..., into the virtual machine.” Ex.
 17 32 at 96. Thus, under the HotLava approach, “protocols and
 18 additional code required to support them can be downloaded and
 19 executed on the fly as needed.” *Id.* at 93. Incorporating into Scout
 the HotLava approach – a Java-based solution as expressly
 proposed in Mosberger – thus clearly satisfies any perceived
 shortcoming of Mosberger with respect to the “dynamically
 identifying” limitation.

20 **Examiner submits that Mosberger in view of HotLava does not**
 21 **appear to disclose “first packet” initiating the “identifying”**
 22 **step of sequence of components. Nowhere within the above**
 23 **cited pages of the Request it is stated that the first packet**
 24 **initiates “identifying” step of sequence of components and all**
 25 **other subsequent “retrieving” step of state information**
 26 **relating to performing processing of previous packet; and the**
 27 **“storing” step of the state information, as required by the**
 28 **claim limitations.**

Therefore, Examiner submits that Mosberger in view of HotLava
 does not appear to disclose... for the **first packet** of the message,
identifying a sequence of components for processing the packets
 of the message... as in claim 1, ...**identifying** a sequence of

components for processing each message based on the **first packet** of the message so that subsequent packets of the message can be processed without re-identifying the components,... as in claim 15 and... **identifying** a message-specific sequence of components for processing the packets of each message upon receiving the **first packet** of the message wherein subsequent packets of the message can use the message-specific sequence identified when the **first packet** was received... as in claim 35.

Since, Mosberger in view of HotLava does not dynamically identify sequences of components based only on the first packet with using pre-defined fields, therefore, proposed rejection of claims 1, 15 and 35, as set forth in pages 263-268 of the Request, does not show a reasonable likelihood that the requester will prevail with respect to at least one of the said claims of the patent. Hence, for the reasons cited above, it is found that the requester has not shown a reasonable likelihood of prevail with respect to claims 1, 15 and 35.

ISSUE 44

Claim 1 states:

...for the **first packet** of the message, **identifying** a sequence of components for processing the packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence; and **storing** an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message...

Similarly, claim 15 states:

...**identifying** a sequence of components for processing each message based on the **first packet** of the message so that subsequent packets of the message can be processed without re-identifying the components,...

Similarly, claim 35 states:

...**identifying** a message-specific sequence of components for processing the packets of each message upon receiving the **first packet** of the message wherein subsequent packets of the message can use the message-specific sequence identified when the **first packet** was received...

In regards to these limitations relating to “**first packet**,” Requester submits in page 269:

The HotLava reference not only renders the claims of the '163 patent when considered in combination with Mosberger, but HotLava also independently and standing alone discloses each and every element of claims 1, 15, and 35. Accordingly, HotLava also fully anticipates these claims for the reasons set forth in detail above, which are incorporated by reference in this proposed ground of rejection.

Examiner submits that HotLava does not appear to **disclose** “first packet” initiating the “identifying” step of sequence of components. Nowhere within the above cited pages of the Request it is stated that the first packet initiates “identifying” step of sequence of components and all other subsequent “retrieving” step of state information relating to performing processing of previous packet; and the “storing” step of the state information, as required by the claim limitations.

Therefore, Examiner submits that HotLava does not appear to disclose... for the **first packet** of the message, **identifying** a sequence of components for processing the packets of the message... as in claim 1, ...**identifying** a sequence of components for processing each message based on the **first packet** of the message so that subsequent packets of the message can be processed without re-identifying the components,... as in claim 15 and ...**identifying** a message-specific sequence of components for processing the packets of each message upon receiving the **first packet** of the message wherein subsequent packets of the message can use the message-specific sequence identified when the **first packet** was received... as in claim 35.

Since, HotLava does not dynamically identify sequences of components based only on the first packet with using pre-defined fields, therefore, proposed rejection of claims 1, 15 and 35, as set forth in pages 263-268 of the Request, does not show a reasonable likelihood that the requester will prevail with respect to at least one of the said claims of the patent. Hence, for the reasons cited above, it is found that the requester has not shown a reasonable likelihood of prevail with respect to claims 1, 15 and 35.

ISSUE 45

Claim 1 states:

...for the **first packet** of the message, **identifying** a sequence of components for processing the packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence; and **storing** an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message...

Similarly, claim 15 states:

...**identifying** a sequence of components for processing each message based on the **first packet** of the message so that subsequent packets of the message can be processed without re-identifying the components,...

Similarly, claim 35 states:

Office Response, '163 Re-exam.

As the Office correctly found, HotLava does not turn on a first packet, does not identify (create) a sequence of components, does not retrieve state information, and does not store state information, as claimed by the patents-in-suit. There is no identification of a sequence based on a first packet.

Interrogatory No. 19:

Identify any use (by Becomm, its agents, or others) of Portal, Strings, or any embodiment of the asserted claims of the patents-in-suit at any time prior to December 29, 1999 (including any connection to or use in of an embodiment in a networked environment, e.g., connection to or use over the Internet, a remote host or network, or any local area or wide area network of any kind), including, individually for each use, a specific description of and date of such use and identification of all related facts, documents, communications, and persons with knowledge.

Response:

1 Implicit refers to and incorporates by reference each of the foregoing General
2 Objections. In addition to the foregoing General Objections, Implicit specifically objects to
3 this interrogatory to the extent it is vague, ambiguous, overly broad, unduly burdensome, and
4 calls for information that is not relevant to the subject matter of this action or to a claim or
5 defense of any party and/or are not reasonably calculated to lead to the discovery of
6 admissible evidence. Plaintiff further objects to this interrogatory on the ground that it seeks
7 information protected by the attorney-client privilege, the work product doctrine, or any other
8 applicable privilege or protective doctrine. Implicit objects further that this interrogatory is
9 duplicative of Juniper's Interrogatory No. 9, and is therefore unduly burdensome. In
10 response to Interrogatory No. 9, Implicit has already identified embodiments of the patents in
11 suit. To the extent that Juniper's current interrogatory seeks to have Implicit identify, beyond
12 this, every "use" such as every instance of "connection to the Internet," including "specific
13 description" of the date, and "all related facts, documents, communications and persons with
14 knowledge," when, moreover, such events would have occurred 13 years ago, is absurdly
15 over-broad and burdensome. Subject to and without waiving these objections, Implicit
16 responds further as follows: Implicit incorporates by reference its response to Interrogatory
17 No. 9.

20 Further, Implicit has made its source code available for Juniper inspection, and
21 Juniper has received that code. Implicit has also produced printouts of key source code; *see*
22 IMP104298-319 ("path.c"); IMP104320-336 ("send.c"); IMP104337-344 ("struct.h"); and
23 IMP104345-55 ("address.c"). Further, the Implicit code base captures the products Implicit
24 built. Implicit will produce this entire base, as against just making it available for inspection,
25 subject to attorneys' eyes only confidentiality.

SUPPLEMENTAL RESPONSE TO INTERROGATORY NO. 19 (6/14/2012)

Implicit further objects to this interrogatory in that it is nonsensical to ask about Implicit “internal disclosures” of its embodiments. If the use is internal, it is not a disclosure. In addition, in Implicit’s extended three and a half day 30(b)(6) deposition, Juniper asked numerous detailed questions concerning Implicit’s internal use of its embodiments.

SUPPLEMENTAL RESPONSE TO INTERROGATORY NO. 19 (7/27/2012)

See also Response to No. 17, above.

Dated: July 27, 2012

Respectfully submitted,

/s/ Spencer Hosie

SPENCER HOSIE (CA Bar No. 101777)
shosie@hosielaw.com
GEORGE F. BISHOP (CA Bar No. 89205)
gbishop@hosielaw.com
DIANE S. RICE (CA Bar No. 118303)
drice@hosielaw.com
HOSIE RICE LLP
Transamerica Pyramid, 34th Floor
600 Montgomery Street
San Francisco, CA 94111
(415) 247-6000 Tel.
(415) 247-6001 Fax

Attorneys for Plaintiff
IMPLICIT NETWORKS, INC.

CERTIFICATE OF SERVICE

I, Jerry Shaw, am a citizen of the United States and am employed in the County of San Francisco, State of California. I am over the age of 18 years and am not a party to the within action. My business address is Hosie Rice LLP, Transamerica Pyramid, 34th Floor, 600 Montgomery Street, San Francisco, California, 94111.

On July 27, 2012, I served the following attached
IMPLICIT NETWORKS, INC.'S SUPPLEMENTAL RESPONSE TO JUNIPER NETWORKS, INC.'S SECOND SET OF INTERROGATORIES (NOS. 11-19)
via electronic mail and U.S. Mail at San Francisco, California, addressed to the following parties:

DAVID C. MCPHIE
dmcphe@irell.com
REBECCA L. CLIFFORD
rclifford@irell.com
Irell & Manella LLP
840 Newport Center Drive, Suite 400
Newport Beach, CA 92660-6324

MORGAN CHU
mchu@irell.com
JONATHAN S. KAGAN
jkagan@irell.com
IRELL & MANELLA LLP
1800 Avenue of the Stars, Suite 900
Los Angeles, CA 90067-4276

*Attorneys for Defendant
Juniper Networks, Inc.*

I certify under penalty of perjury under the laws of the State of California that the foregoing is true and correct.

DATED: July 27, 2012

/s/ Jerry Shaw
Jerry Shaw

EXHIBIT 23
TO BE FILED UNDER SEAL

EXHIBIT 24

1 **THE COURT:** Instead, they've described it as
2 "create."

3 **MR. HOSIE:** Create -- in the context of the reexam,
4 "create" means instantiated.

5 I don't know if this is a fight we can avoid with a
6 claims hearing. And I don't know what, ultimately, at the end
7 of the day, they are going to say it means because they've
8 declined to take a position here.

9 In terms of Mosberger and why Implicit added that
10 term, it meant create as a stateful data processing path.
11 That's what "instantiated" means, the stateful data processing
12 path.

13 In Mosberger, that happened when you plugged the
14 machine into a power outlet. In '163, it happens after the
15 first packet comes in and as a consequence of looking at that
16 graphic, saying, I know what you are and I know what you need
17 so I'm going to build, click click click click click, the data
18 processing path for you right here and now.

19 And you'll recall in our tutorial Mr. Balassanian had
20 an example of '163. The packet comes in. The data flow engine
21 looks at it, and then it snaps together the little modules,
22 red, green, yellow, and whatever. And then and only then does
23 the packet get put through that instantiated data processing
24 path. That's what this word "create" was added to mean, Your
25 Honor.

1 Thank you.

2 **THE COURT:** Thank you.

3 **MR. MCPHIE:** Your Honor, I think we can handle this
4 term fairly quickly with just a couple of slides.

5 **THE COURT:** Tracy, you need to switch it over.

6 **MR. MCPHIE:** I'd like to start where Mr. Hosie
7 started. I believe what I understood him to say was that we
8 should attach great significance to the fact that during the
9 '163 reexamination a word was changed from "form" to "create."
10 If you look at Implicit's proposed construction,
11 "instantiate in memory," it's exactly the same construction for
12 both the word "create," which appears in the '163 patent, and
13 the word "form," which appears in the '857 patent.

14 The parties seem agreed that these two words mean the
15 same thing. And the best explanation of what they mean, I
16 think, Your Honor, can be found in the claims themselves.

17 The question is: How do you create? How do you
18 form?

19 According to Implicit, it's by making a copy of
20 something and putting it in memory.

21 But, in fact, the patent claims answer the question:
22 How do you create? You create by selecting individual
23 components.

24 In other words, how do you create -- how do you form
25 this new sequence that's been created after the first packet

1 was received?

2 Again, it's right in the claim. And it's the same
3 for the '163 patent or the '857 patent.

4 The way you create or form that new sequence is by
5 selecting the individual components in the manner that we
6 showed in the technical tutorial.

7 And, honestly, I don't know that we need to go much
8 further than that, Your Honor. The concept of instantiation is
9 just an attempt to broaden the claim to encompass a very
10 generic concept in electronics, putting data in memory. That
11 clearly is not what Mr. Balassanian claimed in the
12 patents-in-suit.

13 **THE COURT:** Thank you.

14 **MR. HOSIE:** The next term, Your Honor. If we may
15 switch over.

16 "Based on the First Packet of the Message." I'm not
17 going to respond to Mr. McPhie, given our schedule today. And,
18 it's been very thoroughly briefed.

19 "Based on the first packet of the message," we say
20 "plain meaning." I'm always uncomfortable with "plain
21 meaning," but we think this is plain, "based on the first
22 packet of the message."

23 They say, "Based on the format of the data in the
24 first packet of the message."

25 This is another place where they've added that loaded

1 term, given their definition, "format." They've grafted that
2 in there.

3 We don't think the phrase needs construction. We
4 think it's clear. But if it does need construction, it should
5 be relying on information in the first packet. Not one subset
6 narrowly defined. The claim isn't so limited. Says based on
7 the first packet, not based on some subset of information in
8 the first packet.

9 So what the defendants have done here, Your Honor,
10 moving into 18, they've injected a format limitation not
11 present in the claim.

12 We know that's not right, in part because we have
13 other specific claims that do narrow it to using the format.
14 If this wasn't supported by the specification, we couldn't have
15 claims that called for limiting it to the format.

16 I mean, we do say format when we mean format. We
17 didn't in the more general, broader claims.

18 And they also say that we have disclaimed -- this is
19 the material in page 16, that Mr. McPhie said they would stand
20 behind, that we have disavowed using information and headers
21 other than format.

22 That's not true. This is an example. Because we
23 have claims that talk about using information, plurality of
24 information in the header. If it were disclaimed, it couldn't
25 be specifically in the claims.

1 I have nothing further on this, absent questions from
2 the Court.

3 **THE COURT:** Thank you.

4 **MR. HOSIE:** Thank you.

5 **MR. GREEN:** Thank you, Your Honor.

6 Responding for all defendants for the term "Based on
7 the first packet of the message."

8 We disagree, obviously, that this claim term can
9 simply be left as plain meaning. And we also disagree that we
10 can construe it in its general way of simply relying on any
11 information in the packet.

12 If we could swap over to the defendants' claim
13 construction presentation. And if we could move to slide 86.
14 87. All right.

15 So we will agree on one thing, that there is an issue
16 here. And that issue is, when we say "based on the first
17 packet of the message," what is it? Based on what aspect?
18 What -- what part? What is it about the first packet that we
19 are going to base our later system processing on?

20 And the answer to that question is the format of the
21 data. Here's why.

22 Next slide, please.

23 We know from the get-go, looking at this patent,
24 looking at the specification -- again, here we are back in the
25 specification -- the system identifies the message handlers

1 based on the initial data type of the message and a target data
2 type. We're not looking at any information. We're looking at
3 data types and formats.

4 Next slide, please.

5 Again, staying in the specification. This is a
6 conversion system. And it's a conversion system that
7 dynamically identifies conversion routines when data in a
8 source format is received."

9 "Data in a source format." We are looking at the
10 format, not any information in the packet.

11 Next slide, please.

12 The same concept, Your Honor, but this time from the
13 prosecution history, made in an amendment by Implicit to obtain
14 the patent. So these words matter. To use the formats to make
15 decisions, run time decisions, about how to assemble the
16 modules. We are looking at formats, not any information in the
17 packet.

18 Next slide.

19 And maybe this is the point, I hope, that will put up
20 a big punctuation mark on it. If I read Implicit's
21 construction correctly that we want plain meaning, which is
22 sort of a proxy for any information in the packet, we are going
23 to conflate express statements made to obtain this patent in
24 the reexam.

25 So there's three things highlighted. And there's a

1 little bit of connecting to do, so I want to make sure that I
2 lay this out correctly.

3 The first thing is that Mosberger does not identify
4 the sequence based on the first packet of the message. That's
5 Implicit's characterization of Mosberger.

6 So even if it's not correct -- and we are going to
7 have a lot of fights with Implicit later about what Mosberger
8 is or isn't -- we know that Mosberger's at least being used as
9 a proxy here for something that does not, does not identify
10 sequences based on the first packet of the message.

11 We go on, and we see that there's a reference to a
12 page in Mosberger which describes a part of the packet, a part
13 of the packet that the Mosberger system is designed to process.

14 And that part is: "The header of the lowest layer
15 protocol [may]" -- using, again, Implicit's edits -- "include a
16 field that can store the path id." And this path id will be
17 used to perform a demultiplexing operation. And that can be
18 reduced to a simple table lookup.

19 And here's where it closes, jumping down to, "Such a
20 pre-configured path does not satisfy the step of identifying a
21 sequence of components for processing each message based on" --
22 our term of interest here -- "based on the first packet of the
23 message."

24 What do those three things tell us? Mosberger does
25 not identify sequences or process packets based on the first

1 packet. The example, the contradistinction of what Mosberger
2 represents is information from a packet header, a path I.D.,
3 something you could pull out of the structure of the packet and
4 say, oh, here's a piece of information. It tells me which
5 sequence or which path to use to route this packet.

6 That kind of operation is not processing messages
7 based on a first packet of the message.

8 And more specifically to my point earlier about the
9 lookup table, if you look at the next line it's not
10 highlighted, but we see at the very bottom of slide 91 the
11 sentence, "Indeed, such a path by definition avoids any
12 identification of components and instead uses a lookup table to
13 identify a path based on a 'path id.'"

14 All of these things that we're seeing today about the
15 lookup tables being in the claims, about any information from
16 the packet being something you can rely upon, it just doesn't
17 hold water. It's completely contradictory of the prosecution
18 history. It should not be within the scope of the claims.

19 Thank you, Your Honor.

20 **THE COURT:** Thank you.

21 **MR. HOSIE:** "Message" Your Honor, page 22, of the
22 hard copy. We say, "A collection or stream of data that is
23 related in some way." That is explicitly defined in the
24 specification, and that's verbatim.

25 They say -- and more on that in a minute.

1 They say, "A collection of data in a particular
2 format."

3 This is another example where they are ingrafting
4 their narrowing definition of format as one particular bit of
5 information in a packet networking system, in a packet header,
6 to make sure this doesn't cover basic TCP/IP networking.

7 What do they mean by "collection of data in a
8 particular format"? I don't know what they mean, but here's
9 what I think they mean.

10 I think they mean that there's a system where there's
11 some sort of flag on the first packet that says, I am an
12 e-mail.

13 And, Your Honor, you asked Mr. Clark this question.
14 He said, we're going to go from a big e-mail to a little
15 e-mail. Or maybe it was the converse, of little e-mail to a
16 big e-mail. We're going to blow it up. And you said, How
17 would you know? How would the system know? And he said, well,
18 there would be a little flag on the packet that would tell you.

19 That's not how it works. These are nested protocols.
20 It's wrapped. It's like Russian dolls, Your Honor. You know,
21 those stacking Russian dolls. If you look at the outside doll,
22 it doesn't tell you what the innermost doll is. You have to
23 unwrap it. And then you get to the innermost doll and you can,
24 okay, I can see this is a pink and blue doll. You didn't know
25 that until you unwrapped it.

1 That is how TCP/IP works. That is how the Internet
2 works. That is how packets work, Your Honor.

3 This specification uses the word "packet" 51 times.
4 I counted them. 51 times. And they are saying that, you know,
5 it doesn't cover that use given their definition, narrowing
6 definition on format.

7 I really do think this is about the TCP output.
8 That's why they care.

9 TCP outputs all kinds of different stuff. And how
10 does the TCP module know where to send it? It looks at the
11 header where there is a destination and source port, and that
12 tells you.

13 HTTP, by convention, goes to port 80. So if you look
14 at the TCP header, you see port 80. The system knows what to
15 do with it. That's what they're trying to stop.

16 They're saying, we built a bridge to nowhere because
17 you can't get past TCP because it refers to format as a
18 destination port. But, again, "format" broadly defined, output
19 of a protocol. TCP is a protocol. It's covered.

20 "Message" explicitly. I said that we defined it.
21 It's Column 2, line 45 through 47. "A collection of data that
22 is related in some way."

23 It does not have -- it does not have their narrowing
24 in a particular format, whatever that may mean.

25 Thank you, Your Honor.

1 **THE COURT:** Thank you.

2 **MR. MCPHIE:** Your Honor, on "format" it is the
3 plaintiff Implicit that used the word format. It's in the
4 claims.

5 We tried to simplify things. And I understand we are
6 stepping back a little bit to the construction of format. We
7 actually proposed that we jettison all of the dictionary
8 definitions for format and just call it format.

9 Apparently, even using the word format isn't enough.
10 We are not narrowing format in any way, Your Honor. We're
11 trying to apply the terms as they're written.

12 With respect to "message," this is a term that is on
13 the other side of the spectrum from the one we looked at
14 earlier. Sift through carefully and look for clues. There is
15 an express definition in the specification itself.

16 And here's what it is: "A message is a collection of
17 data that is related in some way." So far so good. But the
18 question is "in some way," in what way? Any way?

19 Well, the patent goes on to explain in what way the
20 message packets have to be related. And it does it by way of
21 examples.

22 Such as a stream of video or audio data or an e-mail
23 message. In other words, a collection of packets that are in a
24 common format. Video. Audio. E-mail.

25 Let's skip to this slide here.

1 The problem with "some way," without that explanatory
2 passage that we have included in our construction, Your Honor,
3 is that it opens up the question: Related in what way?

4 Could it be two people in an office building, one is
5 sending an e-mail, one is watching a YouTube video, it happens
6 to go through the same pipe. Is that really related such that
7 we would consider the e-mail and the YouTube part of the same
8 message? It can't be. And here's why.

9 We have here our animation of how to get from blue to
10 red. And this is what the patent does, it puts together that
11 sequence.

12 And it goes through component by component, and it
13 knows that it has to go through a sequence of components to get
14 from the blue to red. The blue is a format, and the red is the
15 output format which is different from the input format.

16 So far so good. But if you can truly have packets in
17 the same message with different formats, what do you do with
18 the orange packet of that message? It doesn't work. You can't
19 put it in the sequence that you've just spent all this time
20 trying to figure out.

21 It's a nonsensical proposition, and the patent
22 nowhere explains how you would go about processing a message
23 with multiple kinds of format of data. No disclosure
24 whatsoever.

25 And so, Your Honor, while we agree those words,

1 exactly the words you see there in the parenthetical, those
2 words accurately describe what a message is, we have -- we've
3 clarified with the words "in a particular format," because it
4 has to be that way.

5 Adding that clarification, it helps the jury
6 understand what way -- what is the way in which the data has to
7 be related? Like a stream of video or audio data or e-mail
8 message.

9 It flows directly from the definition, Your Honor.
10 And that is the way the invention is described, as claimed.

11 **THE COURT:** Thank you.

12 **MR. MCPHIE:** Thank you.

13 **MR. HOSIE:** Finally, Your Honor, "state information."
14 This is the final term of our collection.

15 "State information." Implicit says, "Information
16 specific to a component for a specific message."

17 The defendants say, "Identification of an instance of
18 a specific conversion routine unrelated to the overall
19 sequence."

20 And the rub here is their addition of the limitation
21 "unrelated to the overall sequence."

22 What are they doing here, Your Honor? There is a
23 sentence in the prosecution history where Implicit
24 distinguished Mosberger by saying Mosberger doesn't maintain
25 "state" the way we do. But that was not disclaiming that we

1 maintain state on a component basis that excludes the overall
2 message.

3 You know, again, it's packet-based processing. You
4 take a piece of information. Could be a video. Could be
5 anything. And you chop it up into thousands of tiny little
6 packets, each of which are wrapped with different protocols at
7 each layer. There is no one format for a message. Each packet
8 has multiple formats as you go up and down the layers, if you
9 unwrap and wrap. Okay.

10 As you process that message in '163, you look at the
11 first packet, you figure out what it is, you figure out what it
12 needs, you build your processing path, and then you keep track
13 of the other packets that are related so you don't have to do
14 that whole thing again.

15 The very essence of this system is to avoid the
16 recursive packet-by-packet building a new data path every time.
17 You build it once then you maintain state, which just means
18 track what relates to that message so you can route the rest of
19 the packets belonging to that message with the path that you
20 have built.

21 That's what "maintaining state" means.

22 When they say that it has to be component unrelated
23 to the message, that defeats that entire purpose. It's like
24 saying that -- whoops. It doesn't matter.

25 It's like saying that individual stairs in a

1 staircase are individual so they don't relate to the staircase.
2 Well, they are individual steps, but they clearly relate to the
3 staircase. They are part of it. That's why you have a
4 staircase that gets you from here to there.

5 Thank you, Your Honor.

6 **THE COURT:** Thank you.

7 **MR. MCPHIE:** Your Honor, this is an important term
8 even though we saved it for last, and I want to address, first
9 of all -- we don't have a slide on this, but if you have your
10 patent and can look at Claim 1 of the '163 patent, and I want
11 to address the issue that counsel addressed.

12 Well, the point of the patent is to after you've
13 assembled that sequence, to remember it. So then the next time
14 you get a packet in that message you don't have to go through
15 that laborious process again. That is in the claim. That is
16 part of the invention.

17 And if you look at Claim 1, you'll see it's kind of
18 divided into it two parts. The last sentence of the first half
19 says, "storing an indication of each of the identified
20 components so that the sequence does not need to be
21 reidentified for subsequent packets of the message."

22 That's what Mr. Hosie is talking about. Of course
23 you save that sequence. But that's not what the state
24 information is.

25 Look below at the second half. "For each of a

1 plurality of packets in the messaging sequence," and going on
2 "for each of a plurality of components in the identified
3 sequence." What do you have to do for each of those
4 components? "Receive state information," and further down,
5 "store state information."

6 Both aspects are required, Your Honor. You need to
7 save the sequence that you've put together, true. But each of
8 the individual components also needs to keep track of what's
9 going on. And we tried to illustrate it here.

10 "State information," I think it came up yesterday.
11 State is just a way of saying how you keep track of
12 information; how do you remember information from point to
13 point from time to time?

14 Under the patent as it's claimed, the first half we
15 looked at, that's the first part, keeping that overall memory
16 of the sequence. But the state information is something
17 distinct, and it has to be done for each of the components.
18 And this is the claim language we just looked at.

19 I think the claim language is fairly clear that it
20 has to be done on a component-by-component basis. But to the
21 extent that it -- there's any ambiguity, I believe the
22 prosecution history resolves any ambiguity.

23 Because you see Mosberger, in fact, does keep track
24 of the sequences that overall -- going back here, that top
25 half. Mosberger does do that.

1 And what did Implicit say in response? They said,
2 well, that's really not applicable here. Claim 1 is directed
3 to a method where state information is stored on a
4 component-by-component basis -- and this is the key phrase --
5 and is not information related to an overall path.

6 The question here is not is a stair related to a
7 staircase or is a component related to a sequence? Of course
8 it is.

9 The question is, in this case, the
10 component-by-component information that you're storing, can
11 that just be generalized information that would apply to all
12 the components?

13 It's clear from the claim the answer is no. And they
14 drove the point home during the reexamination proceedings.

15 What we've done, Your Honor, with respect to this
16 construction -- and, I'm sorry. I thought this would have been
17 reflected in plaintiff's presentation as well. This is one of
18 the terms that we attempted to reach some resolution on.

19 And what we did was we said, we will take your
20 proposed construction, and all we're going to do is add this
21 language in blue.

22 That is not related to an overall path. And that
23 language that is not related to an overall path, that comes
24 right out of the prosecution history, Your Honor.

25 "Component-by-component basis." It can't just be something

1 that is related to an overall path.

2 We use that additional language to make clear and to
3 give effect to this disclaimer made in reexamination, which is
4 binding on Implicit.

5 If Your Honor has no other questions, I will
6 conclude.

7 **THE COURT:** All right. Thank you.

8 **MR. MCPHIE:** Thank you.

9 **MR. HOSIE:** If I may very briefly, Your Honor.

10 Mr. McPhie misquoted the patent, I'm sure
11 inadvertently, when he quoted "state information." What it
12 says is, "retrieving" --

13 **THE COURT:** Where are you looking?

14 **MR. HOSIE:** I am looking at '163 reexam, Claim 1,
15 about three-quarters of the way through the claim. He called
16 out "retrieving state information."

17 **THE COURT:** Give me a line number.

18 **MR. HOSIE:** It is line 48.

19 **THE COURT:** Okay.

20 **MR. HOSIE:** Okay. What it really says is "retrieving
21 state information relating to performing the processing of the
22 component with the previous packet of the message."

23 There is clearly an interdependency across components
24 and packets, Your Honor. It's part of the whole process.

25 Finally, in terms of the Mosberger prosecution point,

1 we have addressed this thoroughly on page 14 of the reply
2 brief. I would simply cite the Court to that.

3 There was no intent to disavow some kind of state for
4 the associated packets of a related message. That wouldn't
5 make any sense whatsoever.

6 We have nothing further from the plaintiff, Your
7 Honor, except to thank the Court for its patience and
8 attentiveness.

9 **THE COURT:** Okay. Thank you.

10 **MS. LUTTON:** Your Honor, if I could just have a
11 minute or two of your time. I have one point of clarification
12 and then one summary point that I wanted to make.

13 In terms of the point of clarification, the Court
14 asked Mr. Green yesterday whether dynamically identifying
15 include selecting individual components, if that always has to
16 include. I don't know if the Court remembers that question.

17 I just wanted to be clear for the Court and the
18 record that it does always include, that there is no special
19 meaning for including where it sometimes includes and sometimes
20 doesn't.

21 I can point you to the language.

22 **THE COURT:** Well, I think what I was asking about --
23 perhaps you've addressed it or perhaps it was to a slightly
24 different point.

25 I was thinking about the output of this session,

1 which is supposed to be a claim construction order, which is
2 supposed to single out terms in the claims that may be
3 confusing for the jury because they're not persons skilled in
4 the art, and help them understand what those terms mean. And I
5 was, in that context, asking about what seemed to me to be this
6 cycling around in the definitions that you were requesting be
7 done.

8 Because a further limitation of the claim having to
9 do with "wherein dynamically identifying includes selecting
10 individual components to create the non-predefined (inaudible)
11 of components after the first packet is received," that seemed
12 to me to have been picked up and imported and stuck back in the
13 first half of that same phrase. And I wasn't sure why you were
14 doing that.

15 And when I asked about "including" I probably sounded
16 more knowledgeable than, certainly, I am. What I was thinking
17 of was this, that there are some words you use of art
18 yourselves in the patent world, like "comprising," which means
19 something that it doesn't mean anyplace else.

20 I didn't know if "including" you felt somehow wasn't
21 sufficiently comprehensive that you wanted to pick it up and
22 say it again at the beginning. That's what I was getting at.

23 I don't know if I've answered your question.

24 **MS. LUTTON:** Right. I would say "including" does not
25 have any special meaning. It just means including.

1 **THE COURT:** In contract language, which theoretically
2 we're doing here, "including" really means, well, this is --
3 doesn't mean other things aren't there, but it certainly means
4 this is there. So is that what it means in this context, also?

5 **MR. GREEN:** Your Honor, that's correct. As we read
6 "including" in these claims, it is given its normal function as
7 a term of art in patent drafting, and it includes this thing
8 always, not sometimes and not others.

9 **MS. LUTTON:** But could include more.

10 **MR. GREEN:** Not only this, but at least this.

11 **THE COURT:** Let me cycle back to yesterday, also,
12 which, is nobody answered that question that I was just talking
13 about, which is why do we have to say it twice for the same
14 claim term?

15 **MR. GREEN:** Your Honor, the goal there I think we hit
16 upon when we were addressing the "selecting individual
17 components" term today, is we -- this is one of those things we
18 get in a brawl and we're not sure who swung first, but we
19 proceed and attempt to read the requirement of individual --
20 "selecting individual components" out of the claim altogether.

21 So being careful lawyers we said, yes, selecting
22 individual components is a requirement of this claim. It is
23 included within the things that this claim must do.

24 I think we've addressed that fairly thoroughly today
25 by speaking about "individual components" and what that phrase

1 means, and that "individual" is always there.

2 We also want to be clear that, you know, in
3 addressing that term we didn't want to have words like
4 "identifying" or other parts of that claim read out. So we put
5 it back in and make sure these concepts were preserved.

6 I think I've addressed your question. I don't want
7 to take too much more time or take you off in a direction
8 you're not interested in hearing about.

9 That's why we had the discussion in the context of
10 the bigger phrase yesterday, and then we've addressed many of
11 those points today by focusing on the term "selecting
12 individual components."

13 **MS. LUTTON:** I will just add, Your Honor. We believe
14 that the proper construction of that -- if I can speak for all
15 defendants, but I'm sure I'll see a motion if that's not
16 true -- if the term "includes selecting individual components"
17 is construed properly, we don't, I believe, need it in there
18 twice.

19 Is that correct?

20 **THE COURT:** If it's construed properly --

21 **MS. LUTTON:** If it's construed properly to include
22 the concept of individual components, then we don't need it in
23 there twice.

24 I think that was the judge's concern, was that we
25 were actually listing that term twice.

1 **THE COURT:** Yes.

2 **MS. LUTTON:** And we believe that's unnecessary as
3 long as you don't lose the concept that there are individual
4 components when construing "individual components."

5 One other thing I wanted to address is this concept
6 of TCP/IP has been brought up over and over again.

7 I've got a bachelor's and master's in electrical
8 engineering, and I can't see what we're doing with TCP/IP, that
9 we're supposed to be doing with TCP/IP.

10 It's not something we focused on. Whether there is
11 infringement or not is really going to come down to whether the
12 claim terms as properly construed are infringed by the -- by
13 the defendants.

14 And there's been no focus on our end on TCP/IP. Our
15 focus has been on *Philips*, and this is something I mentioned in
16 the beginning.

17 We believe the proper methodology to be employed here
18 is really looking at the claim specification or the claims and
19 the specification.

20 The prosecution history, whether it's for reexam or
21 otherwise, can be brought in to the extent it's limiting, if
22 they added certain limitations during the prosecution history
23 and they made disclaimers. It can't be used to enlarge the
24 claim terms.

25 So we would just propose to the Court, concerning

1 these terms, follow the (inaudible) analysis, really focus on
2 the claims, the specification, and the prosecution history to
3 the extent that it is a limiting document.

4 **THE COURT:** Thank you.

5 **MS. LUTTON:** Thank you.

6 **THE COURT:** Okay. Thank you very much. The matter
7 is submitted.

8 Do we have a next date right now?

9 **MR. HOSIE:** We do not, Your Honor.

10 (Discussion held off the record)

11 (At 5:25 p.m. the proceedings were adjourned.)

12 - - - -

13
14 **CERTIFICATE OF REPORTER**

15 I certify that the foregoing is a correct transcript
16 from the record of proceedings in the above-entitled matter.

17
18 **DATE:** Thursday, January 26, 2012

19
20 s/b Katherine Powell Sullivan

21 _____
22 Katherine Powell Sullivan, CSR #5812, RPR, CRR
23 U.S. Court Reporter
24
25

EXHIBIT 25
TO BE FILED UNDER SEAL

EXHIBIT 26
TO BE FILED UNDER SEAL

EXHIBIT 27

UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

IMPLICIT NETWORKS, INC.,

Plaintiff,

v.

JUNIPER NETWORKS, INC.,

Defendant.

Case No. C 10-4234 SI

**EXPERT REPORT OF SCOTT NETTLES, PH.D. ON INFRINGEMENT BY JUNIPER
NETWORKS, INC.**

Table of Content

INTRODUCTION	1
SUMMARY OF OPINION	1
EXPERT QUALIFICATIONS	2
REVIEW AND USE OF DOCUMENTS.....	4
CLAIM CONSTRUCTION.....	5
USE OF DEMONSTRATIVES.....	6
LEGAL STANDARDS APPLIED IN THIS REPORT.....	7
LEVEL OF ORDINARY SKILL	9
THE PATENTS AT ISSUE.....	9
A. U.S. Patent 6,629,163	10
B. U.S. Patent 7,711,857	13
TECHNOLOGY BACKGROUND	16
A. A Brief Introduction to Computer Networking	18
STRUCTURE OF THE REPORT	21
THE ACCUSED PRODUCTS	22
ACTS OF INFRINGEMENT BY JNI.....	23
VALUE OF IMPLICIT’S PATENTS (The Value Section).....	34
CONCLUSION.....	41

INTRODUCTION

1. With respect to this report, I have been retained as a technical expert by Plaintiff Implicit Networks, Inc. (“Implicit”) to address the issue of infringement of U.S. Patent No. 6,629,163 (“the ’163 Patent”), and U.S. Patent No. 7,711,857 (“the ’857 Patent”) (collectively “the patents-in-suit”) by certain products of Defendant Juniper Networks, Inc. (“JNI” or “Juniper”).

2. I am being paid for my work in this litigation at the rate of \$450 per hour. My compensation does not depend on the outcome of this litigation. I have no personal interest in the outcome of this litigation.

3. In general, when I say “report”, I mean this main body and all appendices. In some cases I mention the appendices as well as the term “report,” but that is simply for clarity.

4. I reserve the right to modify or supplement my opinion, as well as the bases for my opinion, based on the nature and content of the documentation, data, proof, and other evidence or testimony that the defendants or its expert(s) may present or based on any additional discovery or other information provided to me or found by me in this matter. I expect to testify at trial regarding the matters set forth in this report if asked about these matters by the Court or the parties' attorneys.

SUMMARY OF OPINION

5. As explained in detail in my report, in my opinion a sale, offer for sale, manufacture, exportation, importation, or use of Juniper’s accused products, meets the limitations of the asserted claims 1, 15, and 35 of the ’163 patent and 1, 4, and 10 of the ’857 patent (collectively the “Asserted Claims”).

6. Juniper sold, offered for sale, manufactured, exported, imported and/or used the accused instrumentalities in violation of the patents-in-suit, as discussed below. Juniper’s manufacture, use,

offers for sale, sale, and importation of the accused products each directly infringe the asserted method and computer medium claims.

7. Juniper also indirectly infringes the Asserted Claims of the patents-in-suit. Juniper's manufacture, use, offer for sale, sale, and importation of the accused products, and other acts described herein, are inducing acts in the United States that aid and abet the direct infringement of the Asserted Claims of the patents-in-suit by the users of the accused instrumentalities. As discussed herein, users of the accused instrumentalities perform all the steps of the asserted method claims of the patents-in-suit, and therefore directly infringe the asserted method claims. Because Juniper designed the accused instrumentalities to perform the steps of the asserted method claims, it is my opinion that Juniper's manufacture, use, offer for sale, sale, and importation of the infringing instrumentalities are inducing acts that aid and abet the infringement of the asserted method claims by the end users of the accused instrumentalities.

8. Juniper also indirectly infringes the Asserted Claims of the patents-in-suit by providing hardware and/or software components comprising Juniper's accused instrumentalities that are a material part of the inventions of the Asserted Claims. Further, the accused instrumentalities have been known by Juniper to be made or especially adapted for use in infringement of the patents-in-suit, including in their normal and intended use, and are not staple articles or commodities of commerce suitable for substantial non-infringing use. It is thus my opinion that Juniper has contributed to the infringement of the Asserted Claims.

EXPERT QUALIFICATIONS

9. I have attached a current copy of my curriculum vitae as Exhibit 1. A list of the cases during at least the last five years in which I have signed a Protective Order, have testified as an expert either at a trial, hearing, or deposition, or have submitted statements/opinions is included as Exhibit 1.

10. I attended Michigan State University from 1977 to 1981 as a Merit Scholar and an Alumni Distinguished Scholar, and received a bachelor's degree in Chemistry. I later attended Carnegie Mellon University from 1988 to 1995, during which time I received both a master's degree (1992) and a Ph.D. (1996) in Computer Science. My dissertation was entitled "Safe and Efficient Persistent Heaps" and focused on high performance automatic storage management for advanced database systems.

11. Before earning my Ph.D., I worked for over four years in industry at Silicon Solutions, Inc. and Digital Equipment Corporation, developing computer aided design (CAD) software for the semiconductor and computer sectors. For example, I designed and implemented systems for VLSI mask generation and VLSI design rule checking. I also built the first graphical drawing editor for the X window system, Artemis, which included a sophisticated graphical user interface.

12. I have worked as a professor at three universities since 1995; the University of Pennsylvania, the University of Arizona, and The University of Texas at Austin. I was the recipient of a National Science Foundation CAREER award for "CAREER: Advancing Experimental Computer Science in Storage Management and Education" while I was an Assistant Professor at the University of Pennsylvania. During this time, I also was part of the DARPA funded SwitchWare project, which was one of the pioneering groups in the area of Active Networking ("AN"). My group developed PLAN, the first domain-specific programming language for programmable packets, as well as PLANet, the first purely active inter-network.

13. I joined the faculty of The University of Texas at Austin ("UT"), in the Department of Electrical and Computer Engineering in 1999. In 2005, I was appointed Associate Professor with tenure. At UT, my graduate teaching has focused on networking, including numerous advanced seminars on mobile and wireless networking. My undergraduate teaching has included networking,

operating systems, and one of UT's required programming class, which focuses on programming with abstractions, Java, and data structures.

14. At UT, I continued to develop AN technology and in 2002, my Ph.D. student, Mike Hicks, won the ACM SIGPLAN dissertation award for our joint work on software updating. Along with my Ph.D. student, Seong-kyu Song, I focused my AN work on mobile and wireless networking. As a result, my research shifted away from AN to mobile and wireless networking in general, especially interactions between the network, the radios, and the physical world. Most of my current research involves the development of Hydra, which is a working prototype of an advanced software-implemented WiFi network funded primarily by the NSF.

REVIEW AND USE OF DOCUMENTS

15. In forming the opinions presented in this report, I have reviewed and relied upon among other things:

- The patents-in-suit
- File histories of the patents-in-suit
- All transcripts and exhibits for depositions cited in the report or listed in Exhibit 2.
- All documents cited in this report including all Appendices.
- The Court's Claim Construction Order
- Juniper Source Code
- All documents listed in the technology background section.

16. Unless otherwise noted, the deposition transcripts that I relied upon are final and I have also reviewed the exhibits thereto. In the case that the transcripts are "roughs" or if the exhibits are not yet available, I reserve the right to review the final version and/or exhibits as they become available.

17. In addition, I have attached as Exhibit 2 a list of documents that I have considered in forming the opinions that I express in this report. This list includes documents that I considered independent of whether I relied on them or not.

CLAIM CONSTRUCTION

18. I have been informed that proper infringement analysis begins with determination or construction of the meaning of terms in the Asserted Claims. I understand that the claims are to be construed based upon their ordinary meaning as understood by one of ordinary skill in the art. In forming my opinions on infringement of the Asserted Claims, I have applied the Court's construction to those terms where appropriate. As to the other terms and phrases used in the claims that the Court did not issue a specific construction and for which the parties did not agree to constructions, I have applied the ordinary and accustomed meaning for those terms as would have been understood by one of ordinary skill in the art when viewed in light of the patents-in-suit. I further note that I have replicated the relevant parts of the claim construction below in my claim-by-claim limitation-by-limitation analysis below. Any omissions of terms found there is inadvertent and there is no implication that any missing term was not applied.

20. I understand that the Court has construed the claim terms in the following chart.

Claim Term	Court's Construction
non-predefined sequence of components '163 patent, Claims 1, 15, 35	a sequence of software routines that was not identified before the first packet of a message was received.
dynamically identify[ing] a [message specific] sequence of components '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	Plain meaning
processing [and variants] '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	manipulating data with a program
input/output format '163 patent Claim 1 '857 patent Claims 1, 4, 10	structure or appearance of data to be processed structure or appearance of the data that results from processing
Selecting individual components '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	selecting the individual software routines of the sequence so that the input and output formats of the software routines are compatible
Create/form [the...sequence of components] '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	Plain meaning
based on the first packet of the message '163 patent Claim 15	relying on information in the first packet of the message
Message[s] '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	a collection of data that is related in some way, such as a stream of video or audio data or an email message
State information '163 patent, Claims 1, 15, 35 '857 patent, Claims 1, 4, 10	information specific to a software routine for a specific message that is not information related to an overall path

USE OF DEMONSTRATIVES

21. I reserve the right to make demonstratives (including product demonstrations, product usage, and videos thereof), charts, graphs, or other similar visual aids for trial based upon the opinions

expressed in this report, the data contained in this report, the exhibits or other things cited in this report and/or attached as exhibits to this report.

LEGAL STANDARDS APPLIED IN THIS REPORT

22. I am informed by counsel that the following legal principles apply to the subject matter of this expert report. I understand that the Court will instruct the jury on the law of infringement, and I will follow such instructions. I set forth my understanding of the law of infringement below.

23. I understand that, in general, any person or business entity that makes, imports, uses, sells, or offers to sell within the United States, without permission, any products, systems or methods covered by at least one claim of an asserted patent, may infringe the patent.

24. Burden of Proof on Infringement. I understand that Plaintiff must prove infringement by a preponderance of the evidence (i.e., that infringement is more probably true than not true).

25. When considering whether a person or entity infringes the asserted claims, each of the asserted claims must be considered individually, and to establish infringement of a patent, all of the elements of at least one of the asserted independent claims must be present, either literally or under the doctrine of equivalents.

26. I understand that a patent is literally infringed when the accused product or method meets each limitation of at least one asserted claim.

27. My understanding is that direct infringement of a method patent requires the actual carrying out of the steps.

28. I understand that if an accused device or method does not literally infringe a patent, the device or method may infringe the patent under the doctrine of equivalents.

29. I understand that in addition to direct infringement, infringement liability also arises under indirect infringement. I understand that indirect infringement includes the inducement of

infringement or contributory infringement. I understand that indirect infringement requires direct infringement by a third-party (for example on the part of a customer or other entity).

30. Section 271(c) defines contributory infringement:

Whoever offers to sell or sells within the United States or imports into the United States a component of a patented machine, manufacture, combination or composition, or a material or apparatus for use in practicing a patented process, constituting a material part of the invention, knowing the same to be especially made or especially adapted for use in an infringement of such patent, and not a staple article or commodity of commerce suitable for substantial noninfringing use, shall be liable as a contributory infringer.

31. A party may be liable for contributory infringement if it sells, or offers to sell, a material or apparatus for use in practicing a patented process. That material or apparatus must be a material part of the invention, have no substantial noninfringing uses, and be known to be especially made or especially adapted for use in an infringement of such patent. I understand that Non-infringing uses are substantial when they are not unusual, far-fetched, illusory, impractical, occasional, aberrant, or experimental. In assessing whether a use is substantial, one may consider the use's frequency, the use's practicality, the invention's intended purpose, and the intended market. For a method claim, a party may be liable for inducement or contributory infringement if it sells infringing devices to customers who use them in a way that directly infringes the method claim.

32. I understand informed that to prevail on a theory of inducement of infringement, the patentee must show direct infringement, and that the alleged infringer knowingly induced infringement and possessed specific intent to encourage another's infringement. "Specific intent" can be satisfied by "willful blindness," which is satisfied as: (1) the defendant must subjectively believe that there is a high probability that a fact exists and (2) the defendants must take deliberate actions to avoid learning of that fact."

LEVEL OF ORDINARY SKILL

33. When construing claims of a patent, I understand that it is important for me to consider the level of ordinary skill in the field at the time the invention was made

34. In determining the characteristics of a hypothetical person of ordinary skill in the art of the patents-in-suit at the time of the claimed invention, I considered several things, including the type of problems encountered in the art, the solutions to those problems, the rapidity with which innovations are made, the sophistication of the technology, and the education level of active workers in the field. Finally, I placed myself back in the year 1997, and considered the engineers whom I had taught and with whom I had worked. I came to the conclusion that the characteristics of a person of ordinary skill in the art of the patents-in-suit would be a person with a bachelor's degree in computer science, computer engineering, electrical engineering, or a similar technical field, and at least two years' work experience in those fields.

THE PATENTS AT ISSUE

35. To help introduce the Implicit patents to the jury, I may give a brief description of the parts of the Implicit patents.

36. Like other U.S. patents, the first page of the Implicit patents contains the title of the patent. In addition, the first page lists the inventors and the assignee as well as the abstract and a list of cited references (which may be continued on subsequent pages).

37. Figures cited in the specification are given after the cited references. Following the pages with figures, there are numbered columns in the patent. In the Implicit patents, the numbered columns begin with cross-references to related patent applications.

38. Like other U.S. patents, the Implicit patents follow the cross-references with a section entitled "Background of the Invention" in which the inventors describe the state-of-the art in the year

2000. This is followed in the Implicit patents by a “Summary of the Invention” section. Following this section, a brief description of each of the figures in the patent is given.

39. Next, the inventors give a section entitled “Detailed Description of the Invention” which will include descriptions of “preferred embodiments” of the invention. “Preferred embodiments” are the ways that the inventors believe are the best ways to use or practice their invention.

40. Following the detailed description of the inventions are numbered paragraphs that are the claims of the patents. It is here that the inventors claim their inventions in the Implicit patents.

A. U.S. Patent 6,629,163

41. On December 29, 1999, inventor Edward Balassanian filed the application that issued on September 30, 2003 as United States Patent No. 6,629,163 (“the ’163 patent”) entitled “Method and System for Demultiplexing a First Sequence of Packet Components to Identify Specific Components Wherein Subsequent Components are Processed Without Re-Identifying Components.” On December 18, 2008, a group of companies (led by Intel) put Implicit’s ’163 patent in ex-parte re-exam. After the PTO re-examined the patent, it again found it valid, as amended, and issued a re-examination certificate on June 22, 2010.

42. The asserted claims are listed below:

43. Claim 1 of the ’163 patent reads as follows:

‘163 Claim 1pre

1. A method in a computer system for processing a message having a sequence of packets,

‘163 Claim 1a

the method comprising: providing a plurality of components, each component being a software routine for converting data with an input format into data with an output format;

‘163 Claim 1b

for the first packet of the message, dynamically identifying a non-predefined sequence of components for processing the packets of the message such that the output format of the components of the non-predefined sequence match the input format of the next component in the non-predefined sequence,

‘163 Claim 1c

wherein dynamically identifying includes selecting individual components to create the non-predefined sequence of components after the first packet is received;

‘163 Claim 1d

and storing an indication of each of the identified components so that the non-predefined sequence does not need to be re-identified for subsequent packets of the message;

‘163 Claim 1e

and for each of a plurality of packets of the message in sequence, for each of a plurality of components in the identified non-predefined sequence, retrieving state information relating to performing the processing of the component with the previous packet of the message;

‘163 Claim 1f

performing the processing of the identified component with the packet and the retrieved state information;

‘163 Claim 1g

and storing state information relating to the processing of the component with packet for use when processing the next packet of the message.

44. Claim 15 of the ‘163 patent reads as follows:

‘163 Claim 15pre

15. A method in a computer system for demultiplexing packets of messages,

‘163 Claim 15a

the method comprising: dynamically identifying a non-predefined sequence of components for processing each message based on the first packet of the message so that subsequent packets of the message can be processed without re-identifying the components,

‘163 Claim 15b

wherein different non-predefined sequences of components can be identified for different messages, each component being a software routine,

‘163 Claim 15c

and wherein dynamically identifying includes selecting individual components to create the non-predefined sequence of components;

‘163 Claim 15d

and for each packet of each message, performing the processing of the identified non-predefined sequence of components of the message

‘163 Claim 15e

wherein state information generated by performing the processing of a component for a packet is available to the component when the component processes the next packet of the message.

45. Claim 35 of the ‘163 patent reads as follows:

‘163 Claim 35pre

A computer-readable medium containing instructions for demultiplexing packets of messages,

‘163 Claim 35a

by method comprising: dynamically identifying a message-specific non-predefined sequence of components for processing the packets of each message

‘163 Claim 35b

upon receiving the first packet of the message wherein subsequent packets of the message can use the message-specific non-predefined sequence identified when the first packet was received,

‘163 Claim 35c

and wherein dynamically identifying includes selecting individual components to create the message-specific non-predefined sequence of components;

‘163 Claim 35d

and for each packet of each message, invoking the identified non-predefined sequence of components in sequence to perform the processing of each component for the packet

‘163 Claim 35e

wherein each component saves message-specific state information so that that component can use the saved message-specific state information when the component performs its processing on the next packet of the message.

B. U.S. Patent 7,711,857

46. On October 31, 2007, Edward Balassanian filed a continuation application, which on May 4, 2010, issued as U.S. Patent No. 7,711,857 (“‘857”). Mr. Balassanian assigned the patent to Implicit.

47. The asserted claims are listed below:

48. Claim 1 of the ‘857 patent reads as follows:

‘857 Claim 1pre

1. A method in a computer system for processing packets of a message,

‘857 Claim 1a

the method comprising: receiving a packet of the message and a data type of the message;

‘857 Claim 1b

analyzing the data type of a first packet of the message to dynamically identify a sequence of components for processing a plurality of packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence,

‘857 Claim 1c

wherein analyzing the data type of the first packet of the message to dynamically identify the sequence of components includes selecting individual components to form the sequence of components after the first packet of the message is received;

'857 Claim 1d

storing an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message;

'857 Claim 1e

for each of a plurality of components in the identified sequence: performing the processing of each packet by the identified component;

'857 Claim 1f

and storing state information relating to the processing of the component with the packet for use when processing the next packet of the message.

49. Claim 4 of the '857 patent reads as follows:

'857 Claim 4pre

4. A method in a computer system for processing a message, the message having a plurality of headers,

'857 Claim 4a

the method comprising: analyzing the plurality of headers of a first packet of the message to dynamically identify a sequence of components for processing a plurality of packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence,

'857 Claim 4b

wherein analyzing the plurality of headers of the first packet of the message to dynamically identify the sequence of components includes selecting individual components to form the sequence of components after the first packet of the message is received;

'857 Claim 4c

storing an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message;

'857 Claim 4d

for each of a plurality of components in the identified sequence: performing the processing of each packet by the identified component;

'857 Claim 4e

and storing state information relating to the processing of the component with the packet for use when processing the next packet of the message.

50. Claim 10 of the '857 patent reads as follows:

'857 Claim 10pre

10. A computer readable storage medium, other than a data transmission medium, containing instructions for processing packets of a message, the instructions comprising at least one computer-executable module configured to:

'857 Claim 10a

receive a packet of the message and a data type of the message;

'857 Claim 10b

analyze the data type of a first packet of the message to dynamically identify a sequence of components for processing a plurality of packets of the message such that the output format of the components of the sequence match the input format of the next component in the sequence,

'857 Claim 10c

wherein analyzing the data type of the first packet of the message to dynamically identify the sequence of components includes selecting individual components to form the sequence of components after the first packet of the message is received;

'857 Claim 10d

store an indication of each of the identified components so that the sequence does not need to be re-identified for subsequent packets of the message;

'857 Claim 10e

for each of a plurality of components in the identified sequence: perform the processing of each packet by the identified component;

'857 Claim 10f

and store state information relating to the processing of the component with the packet for use when processing the next packet of the message.

TECHNOLOGY BACKGROUND

51. The subject of the patents-in-suit is broadly computer networking and more specifically the construction of routers and gateways to flexibly and efficiently process messages consisting of related packets. As a result, it will be important that the jury understand certain aspects of networking as background. As an expert in this area, I expect to provide such background to them as part of my testimony. In particular, I would expect to potentially discuss (including but not limited to) such concepts as the network stack, encapsulation, multiplexing and demultiplexing, tunneling, routing, and firewalls. I would expect to provide tutorial information about these and other topics as well as about aspects of networking that are important to the case.

52. I am well suited to provide such tutorial and background information. As shown in my CV (Exhibit 1), I have been active in programming languages, operating systems, and networking research for many years. Networking has been the primary focus of my research since I first joined the faculty of the University of Pennsylvania in 1995, and the majority of my Ph.D. students have written dissertations in the networking area. In addition, I have taught both graduate and undergraduate networking (sometimes both) almost every semester since the spring of 1999 and I have also frequently taught advanced graduate seminars in this area as well. Thus, not only am I knowledgeable about the area, but I am also highly qualified to teach the subject matter.

53. Beyond the material offered as evidence in the rest of this document, I would expect to base my discussion on standard texts (as well as to use them for support of my opinions). In particular, I have read and, in some cases, taught from the following texts, which date both from the time the patents were filed and today:

- Computer Networks, Fifth Edition: A Systems Approach by Larry L. Peterson, Bruce S. Davie Publication Date: March 25, 2011 | ISBN-10: 0123850592 | ISBN-13: 978-0123850591 | Edition: 5
- Computer Networks: A Systems Approach, Second Edition (The Morgan Kaufmann Series in Networking), Larry L. Peterson (Author), Bruce S. Davie (Author) Publication Date: October 25, 1999 | ISBN-10: 1558605142 | ISBN-13: 978-1558605145 | Edition: 2 Sub
- Computer Networking: A Top-Down Approach (6th Edition), James F. Kurose (Author), Keith W. Ross (Author) Publication Date: March 5, 2012 | ISBN-10: 0132856204 | ISBN-13: 978-0132856201 | Edition: 6
- Computer Networking: A Top-Down Approach Featuring the Internet, James F. Kurose (Author), Keith W. Ross (Author) Publication Date: July 10, 2000 | ISBN-10: 0201477114 | ISBN-13: 978-0201477115
- Computer Networks (5th Edition), Andrew S. Tanenbaum (Author), David J. Wetherall (Author) Publication Date: October 7, 2010 | ISBN-10: 0132126958 | ISBN-13: 978-0132126953 | Edition: 5
- Computer Networks, Andrew S. Tanenbaum (Author) Publication Date: March 6, 1996 | ISBN-10: 0133499456 | ISBN-13: 978-0133499452 | Edition: 3rd

54. In addition, the patented technologies are deployed using a wide variety of standard but specialized technologies, which are not treated in depth in standard general textbooks. Thus I have also consulted a number of specialized texts (some of which are cited in other parts of the report) that I

expect to use to support my opinions as well as to help the jury in understanding this complex technology. In particular I expect to use:

- JUNOS Security by Rob Cameron, Brad Woodberg, Patricio Giecco, Timothy Eberhard, James Quinn. Publication Date: September 1, 2010 | ISBN-10: 1449381715 | ISBN-13: 978-1449381714 | Edition: 1
- Junos Enterprise Routing: A Practical Guide to Junos Routing and Certification by Peter Southwick, Doug Marschke, Harry Reynolds. Publication Date: June 25, 2011 | ISBN-10: 1449398634 | ISBN-13: 978-1449398637 | Edition: 2

55. Finally, standards play a very important role in computer networking. Of particular importance to the matters at hand, the Internet Engineering Task Force (IETF) publishes standards in the form of RFC's, which can be found at <http://www.ietf.org/rfc>. These standards form the basis for today's Internet, and define among other things how IP and TCP function. Should a definitive reference about any of the IETF standards used by the Accused Products, I would expect to use these standards.

A. A Brief Introduction to Computer Networking

56. Some basic background is useful to support my opinions below. The basis for this description are the books cited above and my long experience doing networking research and teaching networking.

57. Although there are a wide variety of networking technologies, the Accused Products in this case are designed to be used as part of an internetwork based on IETF standards such as IP and TCP. The "Internet" is the best known example of such a network, but the Accused Products can also play a role in private IP-based "intranetworks" as well. Other standardized networking technology such as

Ethernet also play a role. The discussion below reflects a high-level view of the architecture and implementation of such an internetwork.

58. Modern packet switched computer networks are based on the idea of layering, in which lower layers provide basic functionality which higher layers build on top of and which then even high level layers build upon further. As is the general industry practice, below I have used the numbering scheme from the OSI layering model, although in fact the IETF uses a different (but essentially equivalent at the lower levels) model called the IETF hourglass model.

59. Layer 1 is the “Physical layer.” It is concerned with the actual physical communication of data over electrical wires, optical fibers, radio frequency electromagnetic radiation, or what ever physical transmission medium is used. In the end all communication must pass through the physical layer.

60. Layer 2 is the “Link layer.” It builds upon the Physical layer to allow digital communication of packets of data between two directly connected networking devices. Packets are discrete units of digital data and may be long or short of fixed size or variable (but bounded) size. Ethernet, standardized by the IEEE 802.X standards, is the canonical example. Note Ethernet switch add multihop capability but only among essentially collocated ethernets and so is still considered “layer 2.”

61. Layer 3 is the “Network layer.” It builds upon the Link layers single hop capabilities to provide multihop connectivity. Further it connects networks of different types (e.g. Ethernet and ATM) and can be implemented in such a way to scale across the globe. It provides computer-to-computer communication of packets which typically is “unreliable.” The IETFs Internet Protocol (IP) which underlays the Internet is the best example. We will be concerned with the addresses that IP uses both for the source of a packet and its destination.

62. Layer 4 is the “Transport layer.” It builds on the Network layer to provide a number of possible enhancements. One enhancement is application-to-application communication, using “ports” as addresses. This enhancement is essentially the only one provided by the IETF’s User Datagram Protocol (UDP). The IETF’s Transport Control Protocol supports reliably sending potentially infinite streams of bytes, which it does by breaking the stream up into discrete packets carried by IP. Increasingly the IETF’s Real Time Protocol (RTP) is important for carrying real-time traffic like video or voice on top of IP.

63. Although layers 5-7 are defined by the OSI model, in practice that model breaks down and in general protocols above the transport layer are grouped together under Layer 7, the “application layer.” There is a great deal of complexity here and it does not make sense to generalize. Some well known examples are FTP and HTTP.

64. Note the basic overview above should suffice for the purposes here, but it does ignore issues such as (for example) companion control protocols and shim protocols that are placed between the main layers. I will discuss such complexities as needed below.

65. The next issue concerns how this layering structure is implemented. The problem is that although only the Physical layer can actually communicate, the implementation of each layer on two communicating nodes needs to send control information (that is not part of the data it is sending) to its twin on the other node. To do this it adds a “header” (and sometimes trailer) to the data it is sending that contains this control information. For example, the UDP header contains a source and destination port. Next the layer sends the header plus data to the next lower level, e.g. UDP to IP. The lower layer treats all of this information as data and adds its own header. This process repeats until the physical layer is reached and data is actually sent. This wrapping process is known as “encapsulation” and the

whole process is called “multiplexing.” Thus we see that headers nested inside of headers and so on is an inherent part of IETF-based networks.

66. When the packet arrives at its destination, the whole process must be reversed. The unwrapping aspect is called “de-encapsulation” and the whole process is called “demultiplexing.” Thus we see that when the Accused Products process packets at Layer 3 and above they are demultiplexing them. One further question is during demultiplexing how does the lower level protocol know which of many possible higher level protocols to pass the packet to? The answer is that the header of the lower level protocol has to include some information (a demux key) that’s tells it. For example, Ethernet has an “ether type” field in its header that tells it whether the enclosed data is for IP version 4 or 6 or some other protocol. This demux key along with other control information in the headers comprises the “data type” required by some of the claims.

STRUCTURE OF THE REPORT

67. **My report has four main sections:** 1) this Introduction/Overview section, 2) the Elements section, 3) the Acts section, and 4) the Value section. In the Elements section as described in the Appendix, I address on a claim-by-claim basis, that and how each claim element/limitation is met/found in the accused products. The appendix also contains a technical overview. In the Acts section, I address what activities involving the U.S. constitute acts of infringement under 35 U.S.C. §271. In the Value section, I address the value of the patented technology, from a technological standpoint.

68. In the sections and subsections of my report, I cite to non-exclusive supporting evidence. Given that the asserted claims in the patent-in-suit involve several overlapping claim elements/limitations, the cited evidence has cross applicability. For brevity, this report does not repeat all evidentiary cites across the overlapping claim elements/limitations. However, all cited evidence for

a particular claim element/limitation (or similar claim element/limitation), naturally has cross-applicability, even if not expressly noted.

69. As explained in the Elements section of my report, in my opinion each of the elements of the asserted claims is literally met by the accused products — that is there is literal infringement of all the claims.

THE ACCUSED PRODUCTS

70. JNI's accused products include the following product groups: SRX Services Gateways and J-Series routers, including: SRX100 Services Gateway; SRX210 Services Gateway; SRX220 Services Gateway; SRX240 ServicesGateway; SRX650 Services Gateway; SRX1400 Services Gateway; SRX3400 Services Gateway; SRX3600 Services Gateway; SRX5600 Services Gateway; SRX5800 Services Gateway; J2320 Services Router¹; J2350 Services Router; J4350 Services Router; J6350 Services Router. Junipers sells these products on a modular basis. The accused products listed above include the components sold by Juniper as parts of the product series. Where Juniper sells unpatented components for use with an accused product series listed above, the unpatented components function together with the patented components so as to produce the desired network application delivery product; the components together are analogous to components of a single assembly and complete machine, and constitute a functional unit. For example, for SRX series, the accused product includes the components listed by Juniper as part of that series, *e.g.*, the components listed by Juniper as part of the "SRX 5800" (<http://www.juniper.net/us/en/products-services/security/srx-series/srx5800/>) and

¹ The J-series routers are accused whether or not they include ISM WXC 200.

the parts numbers listed on the “Ordering” page for SRX 5800

(<http://www.juniper.net/us/en/products-services/security/srx-series/srx5800/#ordering>).

ACTS OF INFRINGEMENT BY JNI

71. I have been asked by counsel for Plaintiffs to consider whether JNI directly and/or indirectly infringes the asserted claims with respect to the U.S. in these manners, in light of the technical subject matter in this case. I do so in this section of the report, which I refer to as the “acts section” of my report.

72. In the elements section of this report, I evaluated whether the accused instrumentalities meet the limitations of the asserted claims. In my opinion, they do so, in the manners discussed in that section of my report. In this Acts section of my report, I consider whether Juniper’s acts alone constitute direct infringement, and in conjunction with the acts of others including end-users, constitute indirect infringement by JNI with respect to the U.S. under 35 U.S.C. § 271. In my opinion, Juniper’s acts directly and indirectly infringe the asserted claims, in the manners discussed below in this section of my report and elsewhere.

C. JUNIPER DIRECTLY INFRINGES

73. It is my opinion that JNI infringes the ‘163 and ‘857 patents under 35 U.S.C. § 271(a). JNI’s testing, manufacturer, and sale, and/or importation of the accused products in the United States infringe the ‘163 and ‘857 patents. As described herein, JNI’s accused products meet all the elements of the asserted claims of the patents-in-suit. Its products practice all the elements of the method claims (‘163 claims 1, 15, ‘857 claims 1, 4); they also embody all the elements of the computer media claims (‘163 claim

35 and '857 claim 10). JNI's manufacture, sale and use occurred in the United States, at its own facilities, and at the facilities of customers. Importation into the United States of the accused products, therefore, would also infringe.

74. Juniper practices the methods of each claim in the patents-in-suit. As set forth elsewhere, Juniper's flow-based processing necessarily infringes the patent claims. As described elsewhere, all four-digit SRX devices operate only in infringing, flow-based, mode.² The three-digit SRX and the J-Series operate mostly in flow-based mode. When operated in flow-based mode, they infringe. Thus, the use by Juniper and by its customers and others of the accused products in the flow-based mode infringes the patent claims. Juniper uses the patented inventions (practices each element of each claim asserted here) in activities that include testing its products, selling them and servicing them. Because Juniper's accused products include storage media meeting the limitations of '163 claim 35 and '857 Claim 10, Juniper's manufacture, sale, offering for sale and use of the accused products infringes these claims.³

i. Juniper Uses the Patented Inventions in Internal Testing.

75. Juniper must, and does, thoroughly and extensively test its products. Testing means using the product in the way it was intended, which for the accused products means practicing the patented methods. Like companies in the computer software and hardware industries generally, Juniper extensively tests its products as a

² See e.g. Junos® OS Feature Support Reference for SRX Series and J Series Devices Release 12.1, at 23. http://www.juniper.net/techpubs/en_US/junos12.1/information-products/topic-collections/security/software-all/feature-support-reference/junos-security-feature-support-guide.pdf.

³ I understand that Juniper has been ordered to provide further documentation and testimony concerning use of the accused products in the customer context. I further understand that Juniper produced its witness, Todd Regonin, for deposition on August 14, 2012, but the transcript for this deposition is not yet available. Juniper has not yet produced the ordered documents. I expect to supplement my report when these materials become available.

necessary part of the design, development, modification and release of its products.⁴

Such testing is particularly necessary for companies such as Juniper who provide fundamental computer network infrastructure to business and government, and provide critical network security products. It is essential that its hardware, software, and releases, be thoroughly and extensively tested in development and other contexts prior to sale. Because Juniper's accused products operate entirely (in the case of four-digit SRX) or mostly (in the case of three-digit SRX and J-Series) in flow-based mode, testing will involve practicing the patented methods.

76. As Juniper states: "Each Junos OS build is gated by a full regression run that is fully automated and executes for several days on hundreds of test systems simulating thousands of test cases."⁵ Indeed, "These test cases check for feature functionality, scaling limits, previously known defects and resilience to negative input (such as faulty routing protocol neighbors). *Id.* "Final product will not be shipped" unless the product is tested and passes; each release is so tested:

If a failure occurs in a critical test, the final product will not be shipped until the problem is fixed. This process allows Junos OS releases to occur on a predictable, periodic basis. In fact, many customers trust Junos OS to the point that they run the very first build of each version in production. Still, every Junos OS version is entitled to the so-called regression run (if requested by customers). A regressed release is a fully tested original build with all latest bug fixes applied.

The Junos OS shipping process is based on several guiding principles:

- Every Junos OS release is gated by a systems test, and no releases with service-affecting issues are cleared for shipment.

*Id.*⁶

⁴ <http://www.juniper.net/us/en/local/pdf/whitepapers/2000264-en.pdf>

⁵ <http://www.juniper.net/us/en/local/pdf/whitepapers/2000264-en.pdf>

⁶ Juniper boasts that this testing and release procedure "ensures the exceptional product quality customers have come to expect from Juniper." *Id.* Juniper's "build process occurs simultaneously for all Juniper Networks platforms and uses the same software repository for all products." *Id.*

77. Thus, Juniper tests essentially all aspects of the accused products, including the flow-based functions that are so fundamental to the operation of the accused products.⁷ Juniper also directs testing of, *e.g.*, its SRX firewalls, for purposes of instruction and marketing.⁸

78. Juniper conducts testing at several sites in the United States.
<http://www.juniper.net/us/en/company/careers/job-search/>.⁹

79. Juniper uses its own JUNOS products in its own facilities, including, likely, its SRX line for its security gateway (“we like to use the products that we build ourselves. It’s our preference”). Dyckerhoff Depo. 33-34.

ii. Juniper Practices the Patented Methods in Customer Pre-Sale and Post-Sale Services.

80. Juniper conducts extensive and essential testing and use of the patented method in Juniper’s accused products, both pre- and post-sale, at its customers’ sites in the U.S. and at own U.S. facilities, as described below.

81. Juniper tests its products before it delivers them, including testing them with customers’ configuration information.¹⁰ “Customer-specific testing” takes place “in the test group inside of SBU.” 30(b)(6) Tav. Depo. at 14-15.

⁷ <http://www.juniper.net/techpubs/software/junos-security/junos-security10.0/junos-security-admin-guide/packet-flow-based-fwd-section.html> (“Flow-Based Forwarding . . . Most packet processing occurs in the context of a flow, including management of policies, NAT, zones, most screens, and ALGs”).

⁸ <http://www.juniper.net/us/en/local/pdf/industry-reports/2000482-en.pdf>.

⁹ A Juniper testing engineer will *e.g.*: “Design, develop, and document system level verification tests for Juniper Networks family of routers. Work closely with the software and hardware teams to develop and execute product verification, high availability, and scalability tests for high-performance network routers. Create directed tests based upon examination of system software, hardware designs, and functional specifications.” *Id.*

¹⁰ 30(b)(6) Deposition of Oliver Tavikoli (“30(b)(6) Tav. Depo.”) at 10-15.

Redacted

82. Juniper does “proof of concept” testing, on-site, of the customers’ desired traffic profile and configuration. 30(b)(6) Tav. Depo. at 20-21.¹¹ Redacted

This includes SRX firewall installation. *Id.* at 21-22.¹³

83. Among retail customers, Juniper does “more heavy lifting” in regard to the “testing, lab work, configuration” for customers where Junos is being deployed for the first time. 30(b)(6) Tav. Depo. at 110.

84. Juniper’s Technical Assistance Center (JTAC) provides comprehensive technical assistance and support for customers, through its support engineers, case managers, and support centers.¹⁴ Juniper has JTAC support centers in Sunnyvale, Ogden, UT, Hendon, VA and Westport, CT. *Id.* When a customer experiences a problem and opens a case with JTAC, a Juniper sales engineer will “take ownership of the case” and “make use of all required resources to provide a resolution to the reported problem.” The sales engineer resolution activities include: “review configuration/debug information” for the customer”; “replicate the scenario/issue in the JTAC lab”; and

¹¹ “Proof of concept” is “a lab environment in which you set up something that proves that you can, in fact, do what they [the customer] require you to do.” 30(b)(6) Tav. Depo. at 79.

¹²

Redacted

¹³ The Juniper sales engineers “work with” a customer and “find a way of simulating a load, and they test to see how the machines actually do,” and the engineers and customer personnel “actually operate under test load conditions.” 30(b)(6) Tav. Depo. at 25-27.

¹⁴ <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.

“troubleshoot live on the affected equipment.” *Id.*

Redacted

30(b)(6) Tav.

Depo. at 30-38. JTAC is, post-sale, “where the customer-specific testing would take place.” *Id.* at 14-15.

85.

Redacted

30(b)(6) Tav. Depo. at 66-69.

Redacted

30(b)(6) Tav. Depo. at 70-71.

Redacted

30(b)(6) Tav. Depo. at 70-71.

Redacted

Id. at 67-72.

Redacted

Id. at 72-73.

Redacted

30(b)(6) Tav. Depo. at

75-76.

86. Juniper’s “test group inside of SBU” also does “customer-specific” testing at its facilities (30(b)(6) Tav. Depo. at 14-15), *i.e.*, “customer-level testing . . . inhouse” (*id.* at 93-96). Juniper’s internal testing in its PDT Testing unit (Ronit Polak’s group in Sunnyvale)¹⁵ includes: “As new revisions of code come out for any of the . . . product delivery testing, or PDT, that known configuration that known customers are known to basically be using and to which we want to attest [sic: test] . . . we simply have promised to test those configurations out for them.” 30(b)(6) Tav. Depo. at 95-99. Every time

¹⁵ <http://www.zoominfo.com/#!search/profile/person?personId=1107756000&targetid=profile>.

Juniper has a new release, its PDT tests the release to “capture the essence of the customer’s setup,” including for SRX. *Id.*

D. JUNIPER INDIRECTLY INFRINGES THE ‘163 AND ‘857 PATENTS UNDER 35 U.S.C. § 271(b) AND (c).

87. Juniper infringes the patents-in-suit by inducing its customers and others (“Non-Juniper Direct Infringers”) to infringe. *See* 35 U.S.C. § 271(b).

88. The Non-Juniper Direct Infringers infringe the ‘163 and ‘857 patents by practicing each element of the method claims. As set forth above, the Juniper accused products necessarily infringe, so that when Juniper’s various customers use these products¹⁶, they practice the patented methods. The customers who infringe include those listed above.

89. Juniper actively induces the Non-Juniper Direct Infringers to infringe the ‘163 and ‘857 patents. Juniper intends to cause the acts that constitute direct infringement, as described herein.

90. I have been asked to assume the Juniper had specific intent to induce infringement of the patents-in-suit and/or was willfully blind. Nevertheless, it is my opinion that Juniper knew or should have known that its actions with respect to the Non-Juniper Direct Infringers would lead to actual infringement. At a minimum, Juniper should have known from a review of the ‘163 and ‘857 patents that both itself and the Non-Juniper Direct Infringers were infringing the ‘163 and ‘857 patents. Juniper has put forth no credible non-infringement defenses, and Juniper has not produced any opinion of counsel that would support any belief that its accused products do not infringe the ‘163 and ‘857 patents.

¹⁶ *See e.g.* Matte and Brill depositions.

91. Juniper directs its customers to infringe and instructs them how to do so, describing in great detail how to practice the patented methods. It provides them with the infringing products, directly assists them in practicing the methods, and provides the most detailed instructions, including configurations, in how to infringe. As described above at length, Juniper service personnel work directly with customers to install, implement, configure and test the infringing methods. This involves Juniper instructing its customers, in an extremely detailed way, how to use the accused products in an infringing manner.

92. Juniper provides comprehensive instruction and direction to its customers on how to use and configure its products.¹⁷ This includes a “best-in class Knowledge Base,” which provides users with “thousand s of articles, including configuration assistance, known issues, interoperability, and compatibility information,” and “complete product documentation.” *Id.* Juniper’s JTAC personnel, as described above, will provide service for, *inter alia*, any end user or Juniper Partner with a JTAC technical services or support/maintenance contract.¹⁸

Redacted

... So there’s plenty of SEs [sales engineers] and –and account managers associated with that ... a fair amount of care and feeding and coddling ... that is done with large customers.” 30(b)(6) at 41-42.

Redacted

¹⁷ E.g., <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.

¹⁸ <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.

- Redacted -

93. Juniper provides its customers with “configuration assistance,” through JTAC.²⁰ Redacted

Customer configuration information is shared with Juniper on “a customer-by-customer basis.” 30(b)(6) Tav. Depo. at 36-37. “There’s a certain group of customers that have provided configuration information for their systems.” 30(b)(6) Tav. Depo. at 96. Redacted

¹⁹

Redacted

²⁰ <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>

Redacted

94. Juniper trains its customers in how to use its products, including through its sales engineers and separate training department. 30(b)(6) Tav. Depo. at 39.

95. Juniper provides extensive on-line instruction and training for its products, including SRX and J Series. See <http://www.juniper.net/customers/support/#product>. This includes comprehensive technical documentation²¹ and the Juniper's Knowledge Base.²² These include installation manuals, configuration manuals, feature descriptions, release notes, and other publications describing in detail how to install, configure and operate the Juniper products.^{23 2425}

²¹ <http://www.juniper.net/techpubs/>.

²² See <http://kb.juniper.net/InfoCenter/index?page=home>.

²³ See http://www.juniper.net/techpubs/en_US/junos12.1/information-products/pathway-pages/product/12.1/index.html.

²⁴ Junos® OS Initial Configuration Guide for Security Devices.
http://www.juniper.net/techpubs/en_US/junos/information-products/topic-collections/security/software-all/initial-config/junos-security-swconfig-initial.pdf.

Junos® OS System Basics Configuration Guide Release 12.1.
http://www.juniper.net/techpubs/en_US/junos/information-products/topic-collections/config-guide-system-basics/config-guide-system-basics.pdf.

Junos® OS Feature Support Reference for SRX Series and J Series Devices Release 12.1.
http://www.juniper.net/techpubs/en_US/junos12.1/information-products/topic-collections/security/software-all/feature-support-reference/junos-security-feature-support-guide.pdf.

Branch SRX Series and J Series Selective Packet Services. <http://www.juniper.net/us/en/local/pdf/app-notes/3500192-en.pdf>

JUNOS release notes, http://www.juniper.net/techpubs/en_US/junos12.1/information-products/topic-collections/release-notes/12.1/junos-release-notes-12.1.pdf.

²⁵ As Juniper advises: For a list of related J Series documentation, see <http://www.juniper.net/techpubs/software/junos-jseries/index-main.html>. For a list of related SRX Series documentation, see <http://www.juniper.net/techpubs/hardware/srx-series-main.html>. If the information in the latest release notes differs from the information in the documentation, follow the *Junos OS Release Notes*. To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at <http://www.juniper.net/techpubs/>.

96. Juniper also has training courses for its customers, and provides instruction manuals for its products. 30(b)(6) Tav. Depo. at 114, 124-25.

97. Juniper infringes the patents-in-suit by contributory infringement. *See* 35 U.S.C. § 271(c).

98. The Juniper accused products are a component (or entirety) of the patented machine, manufacture, combination or composition, including the claimed computer media, and a material or apparatus for practicing the patented methods here, and are especially made or especially adapted for use in an infringement of the patents-in-suit, and not staple articles or commodities of commerce suitable for substantial noninfringing use.

99. As described in this report, the accused products meet each element of each claim in the patents-in-suit. They are especially made, by purpose and design, to perform the infringing flow based processing. They are not staple articles of commerce in any sense. The four-digit SRX systems are not suitable for substantial noninfringing use, because, as described in this report, they are only flow-based, and that is the sole means by which the products function. It is also true that the flow-based functionality of all SRX systems and all J-series systems necessarily practices the patented methods, and therefore this functionality has no substantial non-infringing uses.

Juniper Networks supports a technical book program to publish books by Juniper Networks engineers and subject matter experts with book publishers around the world. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration using the Junos operating system (Junos OS) and Juniper Networks devices. In addition, the Juniper Networks Technical Library, published in conjunction with O'Reilly Media, explores improving network security, reliability, and availability using Junos OS configuration techniques. All the books are for sale at technical bookstores and book outlets around the world. The current list can be viewed at <http://www.juniper.net/books>.

HIGHLY CONFIDENTIAL – ATTORNEY EYES ONLY: SOURCE CODE

100. In my opinion Juniper must have known, and did know, that the accused products are especially made or adapted for use in an infringement of the patents-in-suit, and not a staple article or commodity of commerce suitable for substantial noninfringing use. Juniper knew how the products functioned, as set forth above. At a minimum, Juniper should have known from a review of the '163 and '857 patents that both itself and the Non-Juniper Direct Infringers were infringing the '163 and '857 patents. Juniper has put forth no credible non-infringement defenses, and Juniper has not produced any opinion of counsel that would support any belief that its accused products do not infringe the '163 and '857 patents.

VALUE OF IMPLICIT'S PATENTS (The Value Section)

101. The network packet processing technology invented by Implicit has significant value to not only Implicit, but also to users and purchasers of Implicit's technology.

102. The infringing, flow-based, functionality is a fundamental feature of Juniper's SRX gateways and J-Series router product lines, the accused products in this case. Juniper's four-digit SRX Gateways operate entirely in infringing flow-based mode. Thus, if they did not practice the patented methods, and embody the patented computer media, they simply would not function.

103. The infringing, flow-based, functionality is also a fundamental feature of Juniper's three-digit SRX gateways and J-Series routers. While the three-digit SRX Gateways and J-Series router can operate in a non-infringing packet-by-packet manner, many key functions can be performed by these products only in flow-based mode, and,

according to JNI, “most packet processing occurs in the context of a flow,” *i.e.*,
infringing flow-based processing.²⁶

104. Further evidence presented below shows that the infringing flow-based functioning is fundamental for the accused products to operate in the manner in which JNI intends, and in which the products are used by customers.

105. Juniper’s four-digit SRX Gateways (SRX1400, SRX3400, SRX3600, SRX5600, SRX 5800) are the top of Juniper’s line, providing network gateway services for extremely large and sophisticated customers, for data center and other uses, in industries such as telecommunications and financial services which demand the most sophisticated and advanced network security. These SRX Gateways operate only in flow-based mode, not packet-based. As testified-to by a Juniper designee, Juniper’s four digit devices “provide just the flow-based processing.” Narayanaswamy Depo. at 123-24.27 When it comes to Juniper’s top of the line models, infringing, flow-based processing, is the way the accused products perform network processing. These gateways operate in flow-based mode because the technological state of the art, and Juniper’s customers, demand it, as discussed below.

106. While SRX three-digit gateways and J-series routers are capable of packet-based processing, “most packet processing occurs in the context of a flow,” *i.e.*,

²⁶ JUNOS Software Security Configuration Guide (June 7, 2012) <http://www.juniper.net/techpubs/software/junos-security/junos-security10.1/junos-security-swconfig-security/junos-security-swconfig-security.pdf>, at 4-5, 45. See also <http://www.juniper.net/techpubs/software/junos-security/junos-security10.0/junos-security-admin-guide/junos-security-admin-guide.pdf>, at 271 (“Most packet processing occurs in the context of a flow, including management of policies, NAT, zones, most screens, and ALGs”).

²⁷ See also http://www.juniper.net/techpubs/en_US/junos12.1/information-products/topic-collections/security/software-all/feature-support-reference/junos-security-feature-support-guide.pdf at 25.

“flow-based processing.”²⁸ For the key security function, stateful flow-based processing, is essential for modern security services. “Stateless security sucks,” as Juniper’s designee stated. 30(b)(6) Deposition of Oliver Tavakoli (“30(b)(6) Tav. Depo.”) at 52.

107. As Juniper states for the J Series:

Flows and Sessions

Flow-based packet processing, which is stateful, requires the creation of sessions. A session is created, based on the characteristics assessed for the first packet of a flow, for the following purposes:

- To store the security measures to be applied to the packets of the flow
- To cache information about the state of the flow

For example, logging and counting information for a flow is cached in its session. (Some stateful firewall screens rely on threshold values that pertain to individual sessions or across all sessions.)

- To allocate required resources for the flow for features such as Network Address Translation (NAT) and IPsec tunnels
- To provide a framework for features such as Application Layer Gateways (ALGs) and firewall features

Most packet processing occurs in the context of a flow. The flow engine and session bring together the following features and events that affect a packet as it undergoes flow-based processing:

- Flow-based forwarding
- Session management, including session aging and changes in routes, policy, and interfaces
- Management of virtual private networks (VPNs), ALGs, and authentication
- Management of policies, NAT, zones, and screens²⁹

108. Thus, although three-digit SRX and J routers can perform in packet-based mode, “by default, Juniper Networks devices running Junos OS use flow-based

²⁸ JUNOS Software Security Configuration Guide (June 7, 2012) <http://www.juniper.net/techpubs/software/junos-security/junos-security10.1/junos-security-swconfig-security/junos-security-swconfig-security.pdf>, at 4-5, 45. See also <http://www.juniper.net/techpubs/software/junos-security/junos-security10.0/junos-security-admin-guide/junos-security-admin-guide.pdf>, at 271 (“Most packet processing occurs in the context of a flow, including management of policies, NAT, zones, most screens, and ALGs”).

²⁹ JUNOS Software Security Configuration Guide (June 7, 2012) <http://www.juniper.net/techpubs/software/junos-security/junos-security10.1/junos-security-swconfig-security/junos-security-swconfig-security.pdf>, at 45.

forwarding.”³⁰ Key functions such as stateful firewalls, IKE, IPSec and IPS can be performed only in flow-based mode. 30(b)(6) Tav. Depo. at 105-109. “All J-series routers and SRX Services Gateways are shipped from the factory in a secure [stateful] context.”³¹ The reasons for this include: “Firewalls and security devices take a session-based processing approach. Session-based or flow-mode processing leverages session state to minimize packet-by-packet decision making, and this improves the overall performance of SRX Series Services Gateways for the branch. In flow mode, traffic is inspected at the transport level using a five-tuple match of source and destination addresses, source and destination ports (when applicable), and protocol with the source and destination zones to determine if the packet belongs to a new or existing session.”³² Compared to packet-based processing, “[f]low-mode forwarding, however, allows for more granular traffic control.” *Id.*

109. For example, the following services benefit when sessions are monitored:

- Stateful inspection
- NAT
- Intrusion prevention system (IPS)
- Unified threat management (UTM) (such as antivirus, content filtering, Web filtering, antispam, etc.)
- J-Flow

³⁰ http://www.juniper.net/techpubs/en_US/junos12.1/topics/concept/security-selective-stateless-packet-based-service-understanding.html.

³¹ Junos Enterprise Routing, Juniper Networks Technical Library (O'Reilly 2011) at 601-02.

³² <http://www.juniper.net/us/en/local/pdf/app-notes/3500192-en.pdf>.

*Id.*³³

110. The movement to flow-based processing was a major advance in computer network application delivery and security. This is underscored by the fact that not only Juniper but other leading companies, such as F5 and others,³⁴ moved to flow-based processing. Juniper itself recognizes that while Juniper routers had historically been limited to packet-based processing, a “primary change” in JUNOS was “the addition of flow-based processing.”³⁵ “Handling traffic as flows offers significant benefits for stateful services.”³⁶ In the current era, stateless firewalls simply are not an acceptable substitute for stateful, flow-based firewalls. “Stateless security sucks.” (30(b)(6) Tav. Depo. at 52. IDP/IPS and IPSec for SRX “are implemented in a flow-based mode, and they’re not in the packet-based mode.” (30(b)(6) Tav. Depo. at 105-106.

111.

Redacted

³³ Flow-based forwarding in the branch SRX and J series also provide routing and a variety of other services. *Id.* at 5.

³⁴ Additional evidence of the critical significance of the patented invention to networking products such as Juniper’s accused products is presented in my report provided with respect to F5 Networks. That report is incorporated herein by reference.

³⁵ Junos Enterprise Routing, Juniper Networks Technical Library (O’Reilly 2011) at 597-8.

³⁶ Junos Enterprise Routing, Juniper Networks Technical Library (O’Reilly 2011) at 597.

112. Juniper's telecommunications customers also buy SRX because of their stateful, flow-based, functionality. Redacted

at 49. If "you're going to provide . . . stateful security," the product has "got to maintain . . . a flow table session." Redacted

at 53-
54.37

113. Redacted

114. Juniper's large retail customers for its 3-digit SRX boxes also use them only in flow-based mode. Beyond the financial services and telecommunications customers, Mr. Tavakoli identified as among Juniper's largest SRX customers Redacted

(30(b)(6) Tav. Depo. at 105. It does so because it is interested in "features that would only be available in flow-based mode," IKE, IPSec and IPS. *Id.*

³⁷ The price of a single 5K SRX box is "north of a million" dollars. Tav. Depo. at 100.

at 105-106. “These are things on our box that are implemented in a flow-based mode, and they’re not in the packet-based mode.” *Id.*

Redacted

(30(b)(6) Tav. Depo.

at 108-109.

115. It is my opinion that there are no acceptable non-infringing alternatives. My understanding is that any acceptable non-infringing alternative must provide the same or equivalent functionality as provided by the patented technology. Here that means stateful, flow-based processing in which a plurality of components can be combined at the time of the arrival of the first packet of a message. This functionality is both powerful and flexible. I note that approaches such as stateless firewalling are not acceptable non-infringing alternatives because they do not provide equivalent function. I understand that acceptable may be based on many criteria, cost to implement, cost to the customer, performance, and so on. In my opinion, the critical one here is performance. Juniper’s Accused Product’s compete in a marketplace where both price/performance and absolute performance are very important. The patented technology as used by Juniper provides high performance while still providing the required functionality by remembering (or “caching”) the results of setting up the flexible, stateful processing path created upon receipt of the first packet. This is central to the infringement. In my opinion, repeating this computation for each packet would result in unacceptable performance. In fact, this performance issue is one that has been long known to the networking community and, in my opinion, contributed to the slow acceptance of flow-

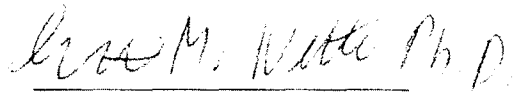
based techniques. I know of no alternative to caching the results of the first packet computation that would provide acceptable performance.

116. Finally, Juniper has not disclosed any proposed alternatives. Should they do so in the future, I reserve the right to rebut any such proposal.

CONCLUSION

117. As discussed throughout my report, it is my opinion that Juniper infringes claims 1, 15, and 35 of the '163 patent and 1, 4, and 10 of the '857 patent.

118. I reserve the right to amend or supplement my opinions upon any future finding by the Court regarding claim construction, or receipt of any additional information.

A handwritten signature in cursive script, reading "Scott M. Nettles Ph.D.", written in dark ink. The signature is positioned above a horizontal line.

August 15, 2012

Scott M. Nettles Ph.D.

EXHIBIT 28
TO BE FILED UNDER SEAL